НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ імені ІГОРЯ СІКОРСЬКОГО»

Факультет прикладної математики

Кафедра програмного забезпечення комп'ютерних систем

«На правах рукопису»
УДК _____

«До захисту допущено»
Науковий керівник кафедри
_____ І.А. Дичка
«___»_____2018р.

Магістерська дисертація

на здобуття ступеня магістра

зі спеціальності 121 Інженерія програмного забезпечення

на тему: «Метод автоматизованої класифікації коротких новинних текстів з використанням іменованих сутностей»

Виконав:
студент VI курсу, групи КП-61м
Бартков'як Андрій Юліанович _____

Керівник:
Доцент кафедри ПЗКС, к.т.н., доцент,
Заболотня Т.М. _____

Рецензент:
Доцент кафедри ММСА ІПСА, к.т.н., доцент,
Дідковська М.В. _____

Засвідчую, що у цій магістерській дисертації немає запозичень з праць інших авторів без відповідних посилань.
Студент _____

Київ – 2018 року

ЗМІСТ

ВСТУП

Поточний стан інформатизації суспільства характеризується, в першу чергу, масовим залученням Інтернет до всіх сфер діяльності людини. Можливості всесвітньої мережі широко використовуються для роботи та розваг, для шопінгу, бронювання та покупки квитків, спілкування, отримання інформації про новини. Наразі існує безліч новинних порталів та платформ, які пропонують увазі користувача новини на будь-яку тематику.

Сьогодні звичайна людина отримує кожен день у п’ять разів більше інформації, ніж 30 років тому [1]. Це легко пояснюється тим, що дізнатися найсвіжіші новини тепер можна не лише з газет, журналів, телебачення чи радіо, а й з Інтернету. Саме Інтернет посідає чільне місце серед джерел, з яких можна дізнатися свіжі новини. Проте, на теренах всесвітньої мережі пропонується занадто широкий та не завжди правильно впорядкований набір новинних текстів, і загубитися в цьому морі інформації дуже просто.

Окрім широкого спектру новинних порталів та сайтів Інтернет породив нові тенденції у медіа-просторі. Так широкої популярності набули короткі новинні тексти, які дозволяють швидко ознайомитися з суттю новини, не вдаючись в непотрібні деталі. Такий формат новин дозволяє залишатися в курсі подій, значно скорочує час пошуку та ознайомлення з новиною, а також дозволяє охопити значно більшу кількість інформації, ніж класичні новинні джерела.

Однією з найбільш популярних соціальних мереж, якою користуються для публікації коротких новинних текстів, є Twitter. Станом на 2017 рік дана соціальна мережа охоплює близько 330 мільйонів активних користувачів [2]. Більшість поважних новинних медіа мають офіційні аккаунти в соціальній мережі Twitter та публікують там короткі новини, які подані в більш повній формі на їх сайтах. Даний підхід значно спростив вибір необхідної новини для прочитання.

Проте в останні роки стрімке зростання чисельності Інтернет-аудиторії та відповідне збільшення кількості коротких новинних текстів та так званих «твітів» користувачів Twitter відчутно ускладнило задачу формування найбільш релевантної новинної стрічки та призвело до збільшення часових витрат на її обробку. Користувачі, що підписані на велику кількість сторінок новинних медіа, часом можуть не встигати за оновленням своєї новинної стрічки, що взагалі унеможливлює ручний аналіз вмісту твітів.

Таким чином, розроблення методів, алгоритмів, а також програмних засобів автоматизованої класифікації коротких новинних текстів є актуальною задачею на сьогоднішній день.

1 ОГЛЯД МЕТОДІВ АВТОМАТИЗОВАНОЇ КЛАСИФІКАЦІЇ ПРИРОДОМОВНИХ ТЕКСТОВИХ ДАНИХ

1.1 Огляд існуючих підходів до автоматизованої класифікації текстових даних

Класифікація – це процес віднесення об’єкта класифікації до одного з ряду різних наборів або категорій – класів згідно певному правилу класифікації.

Правило класифікації є процедурою, згідно з якою кожна з визначених елементів належить до одного з класів [3]. Алгоритм, який реалізує класифікацію, особливо в конкретній реалізації, відомий як класифікатор. Термін «класифікатор» іноді також відноситься до математичної функції, виконаної за алгоритмом класифікації, який відображає вхідні дані до категорії.

Клас – це сукупність об’єктів, які можуть бути однозначно визначені властивістю, яку поділяють всі її учасники.

Класифікацію можна розглядати як дві окремі проблеми – бінарну класифікацію та багатокласову (мультикласову) класифікацію.

Бінарна або двійкова класифікація – це одна з підзадач класифікації, що полягає у віднесенні елементу до заданого набору, що складається з двох класів (передбачається, що кожен елемент належить до однієї з двох груп), заснованому на заданому класифікаційному правилі.

Багатокласова або мультикласова класифікація – це одна з підзадач класифікації, що полягає у віднесенні елементу до одного з трьох або більше класів [4].

Багато класову класифікацію не слід плутати з класифікацією з декількома мітками (лейбами), де для кожного екземпляру слід передбачити декілька міток.

Класифікація з декількома мітками – це одна з підзадач класифікації, що полягає у віднесенні кожного екземпляру до одного або більше класів,

тобто присвоєнні цьому класу декількох міток. Класифікація з декількома мітками є узагальненням багато класової класифікації, яка є одномітковою задачею класифікації сутностей з використанням більше ніж двох класів. В задачі з декількома мітками не існує жодних обмежень щодо кількості класів, до яких може бути присвоєний екземпляр.

Формально багатоміткова класифікація це проблема пошуку відповідності вхідного елементу x до бінарного вектору y (присвоєння значення 0 або 1 кожному елементу (мітці) в y).

Класифікація документів – одне із завдань класифікації, яка полягає у віднесенні документа до однієї з декількох категорій на підставі змісту документа. Документи, що підлягають класифікації, можуть представляти з себе текст, зображення, музику тощо. Кожен вид документа має свої особливі класифікаційні проблеми. Під класифікацією документів передбачається класифікація тексту, якщо не вказано інше.

Класифікація документів може бути вміст або на запит.

Класифікація на основі вмісту – це класифікація, в якій вага, присвоєна окремим частинам документу, визначає клас, до якого призначається документ. Наприклад, існує загальне правило для класифікації в бібліотеках, що принаймні 20% вмісту книги має бути приблизно в класі, до якого присвоюється книга [5]. В автоматизованій класифікації метрикою вмісту може бути кількість разів, коли слова з’являються в документі.

Класифікація, орієнтована на запити (або -індексування), є класифікацією, в якій очікуваний запит від користувачів впливає на те, як документи класифікуються. Перед класифікатором стоїть завдання за якими дескрипторами слід знаходити цю сутність та врахувати всі можливі запити і визначити, для яких запитів сутність є релевантною [6].

Класифікація може бути проведеною «вручну» (ще таку класифікацію називають «мануальною», «інтелектуальною») або алгоритмічно. Інтелектуальна класифікація документів в основному

застосовується у галузі бібліотекознавства, проте її використання в інших галузях теж є досить широким. Так, у класифікації новинних текстів клас («жанр», «тип») статті теж часто визначається вручну автором статті. Новини розміщені на веб-сайтах сайтах або друкованих виданнях зазвичай віднесені до певної категорії адміністраторами, що розміщують публікацію. Проте така класифікація відбувається не завжди, крім того дані мітки про новину не зберігаються при публікації в соціальних мережах – місці, де досить розповсюдженим є використання коротких новинних текстів. Зважаючи на вищесказане в контексті даного дослідження будуть розглядатися лише автоматизовані підходи до класифікації коротких новинних текстів.

Загалом існуючі підходи до автомазованої класифікації природномовних текстових даних в цілому та класифікації новинних текстів зокрема можна розділити на наступні категорії [7].

Підходи, засновані на використанні правил

Даний підхід базується на наборі правил, використовуючи які система робить висновок про те, що якого класу відноситься даний текст.

Перевагами даного підходу є:

- Велика точність при наявності якісного набору правил.
- Надає найкращу точність, якщо зосереджений на певній вузькій тематиці.

У якості недоліків можна визначити:

- Підхід вимагає великих ресурсних затрат на створення правил.
- Підхід не є універсальним.

Підходи, засновані на використанні словників

В даному підході використовують спеціалізовані словники. Кожне слово з тексту заміняється його вагою з словнику і потім вираховують загальну вагу тексту. «Вага» вказує до якого класу можна віднести текстовий документ. Для підрахунку загальної ваги в найпростішому випадку використовують пошук середнього арифметичного [8].

Серед переваг даного підходу можна визначити:

- Відносно високу точність на певній обраній тематиці.
- Простоту програмної реалізації.

Недоліками є:

- Висока ресурсоємність задачі створення словників.
- Підхід не є універсальним.

Машинне навчання з учителем

Головною суттю таких методів є те, що на першому етапі навчається машинний класифікатор на заздалегідь розмічених текстах, а потім використовують отриману модель (зазвичай векторну) при аналізі нових документів.

Серед переваг можна виділити:

- Висока точність.
- Метод є відносно універсальним.

Недоліками є:

- Потреба пошуку та підготовки матеріалів для навчання.
- Складнощі у перенавчанні методу.

Машинне навчання без учителя

В основі цього підходу лежить ідея, що терміни, які найчастіше зустрічаються в цьому тексті і в той же час присутні в невеликій кількості текстів у всій колекції мають найбільшу вагу в тексті. Виділивши ці

терміни, а потім визначивши їх клас, можна зробити висновок про клас, до якого належить весь тексту в цілому [9].

Перевагами даного підходу є:

- Відсутність потреби створення словників або набору даних для навчання.

У якості недоліків можна визначити:

- Алгоритми, що засновані на даному підході, на даний момент надають низьку точність.

Дані підходи до автоматизованої класифікації текстових даних мають як переваги, так і певні недоліки. Наприклад, класифікація, що базується на використанні правил, вимагає великих витрат часу та людських ресурсів на розробку, тому що для отримання високої точності при роботі відповідної програмної системи, необхідно створити велику кількість правил. Недоліком підходів, що засновані на використанні словників, є потреба у створенні цих словників, що теж є трудомісткою задачею, особливо у контексті класифікації новинних текстів (адже не існує жорсткого ієрархії за якою здійснюється класифікація новинних текстів). Методи, основані на машинному навчанні без учителя, показують найменш точні результати визначення класу вхідних текстових даних.

Зважаючи на наведені вище недоліки, в межах даної роботи було сконцентровано увагу на дослідженні існуючих методів класифікації природномовних текстових даних (а саме новинних текстів), в основі яких лежить машинне навчанням з учителем, оскільки дані методи є досить універсальними, дають непогану точність отримуваних результатів та не є ресурснозатратними.

1.2 Аналіз особливостей класифікації коротких новинних текстів

Новина – оперативне інформаційне повідомлення, яке містить суспільно важливу та актуальну інформацію, також стосується певної

сфери життя суспільства загалом чи окремих його груп. Новини розповсюджуються за допомогою ЗМІ.

ЗМІ чи новині медіа – це засоби масової інформації, які спрямовані на донесення новин до широкої громадськості або цільової аудиторії.

Новини поширюються завдяки різним видам засобів масової інформації, таких як:

- преса (друковані видання такі як газети, журнали, альманахи, збірки);
- радіо;
- телебачення;
- Інтернет (Інтернет-газети, новинні блоги та ін.).

У даному дослідження увагу було зосереджено на новинах, що розповсюджуються за допомогою Інтернет-видань. Популяризація Інтернету серед більшості населення планети [10] спонукало традиційні засоби масової інформації та мас-медіа розвиватися та шукати нові підходи до читачів, щоб продовжувати виконувати своє основне завдання – оперативно розповсюджувати актуальну інформацію. Це стало можливим завдяки телефонам, планшетам, ноутбукам, які завжди під рукою, а об’єднує всі ці гаджети мобільність та можливість доступу до всесвітньої мережі. Це спричинило появу такого явища як Інтернет-журналістика – це підвид журналістики, що випускається чи розповсюджується через Інтернет. Все почалося з відкриттів великими виданнями відповідних сайтів у мережі Інтернет, що в більшості своїй відтворювала зміст друкованого видання на сторінці веб-сайту. Таким чином багато новинних організацій, розташованих в інших засобах масової інформації, також стали поширювати новини в Інтернеті. Наприклад, BBC News, Reuters, CNN, Mirror, The Guardian, Deutsche Welle, Aljazeera, Bloomberg – це далеко не вичерпний перелік засобів масової інформації, які активно ведуть свою діяльність в мережі Інтернет.

Інтернет дозволив офіційну та неформальну публікацію новин через основні засоби масової інформації, а також блоги та інші само опубліковані новини. Так еволюція мас-медіа у середовищі всесвітньої павутини призвела до появи нового їх втілення – соціальних медіа. Соціальні медіа (англ. Social media) – вид мас-медіа, завдяки якому споживачі контенту через свої дописи стають його співавторами і можуть взаємодіяти, співпрацювати, спілкуватися, ділитися інформацією або брати участь у будь-якій інший соціальній активності з теоретично усіма іншими користувачами певного сервісу.

Таким чином, окрім веб-сайтів, класичні засоби масової інформації почали створювати в соціальних мережах сторінки, які дали змогу розповсюджувати новини без потреби у переході на веб-сайт. Проте, частіше за все в соціальних мережах публікуються короткі новинні тексти, які здобули широку популярність серед Інтернет-аудиторії завдяки своїй лаконічності та інформативності. Зазвичай короткі новинні тексти складаються з назви та анотації. Такі вид новин дозволяє зекономити час та уникати новин, теми яких є заздалегідь не цікавими користувачеві.

Звичайно існують випадки клікбейтних новин. Клікбейт (англ. Clickbait, від click – клацання і bait – наживка) – це посилання на веб-сторінку, призначене для того, щоб змусити користувачів переходити на певну веб-сторінку чи відео. Зазвичай клікбейтні заголовки спрямовані на використання так званого «curiosity gap», забезпечуючи достатньо інформації, щоб зацікавити читачів, але недостатньо, щоб задовольнити їх цікавість, не натискаючи посилання на пов’язаний вміст [11] [12] [13]. По суті своїй методи, використані авторами клікбейтних новин, можна вважати похідними від жовтої журналістики, яка представляла мало або взагалі не правдиві новини, а замість цього використовувала приголомшливі заголовки, які включали перебільшення новинних подій, скандальних переслідувань або сенсацій [14]. В основі клікбейтингу лежить технологія гіпертексту, саме вона дозволяє робити легкі переходи

між веб сторінками по кліку. Гіпертекст – це текст, який відображається на дисплеї комп'ютера або інших електронних пристроях з посиланнями (гіперпосиланнями) на інший текст, який читач може негайно отримати, або де текст може розкриватися поступово на декількох рівнях деталізації (також називається StretchText) [15]. Гіпертекстові документи взаємопов'язані гіперпосиланнями, які, як правило, активуються клацанням миші, натисканням певної клавіші на клавіатурі або натисканням на екрана, що реагує на дотик. Крім тексту, термін «гіпертекст» іноді також використовується для опису таблиць, зображень та інших форматів представницького вмісту з вбудованими гіперпосиланнями.

Але так як клікбейт є своєрідною трансформацією з друкованих видань так званої жовтої преси у площину Інтернет видань, і коротких новинних текстів, то можна з упевненістю сказати, що частка клікбейтних новин приблизно відповідає частці жовтих видань серед серйозної преси. Саме тому у контексті даного дослідження автор не зосереджувався на дослідженні клікбейту.

Суттєвою відмінністю коротких новинних текстів є те, що зменшення об'єму тексту може ускладнити визначення класу, якого стосується новина. Інколи заголовки та анотації взагалі можуть не нести жодного змісту, якщо не знати контекст, який знаходиться поза межами новини. Наприклад, «Деніел Редкліфф відмовляється повертатися до свого персонажу, який приніс йому славу, любов всіх дітей планети та високі статки.». Дана коротка новина містить у собі 128 символів, що підходить до ліміту повідомлення в соціальній мережі Твітер (140 символів). Не знаючи, що Деніел Редкліфф – актор, а персонаж – Гаррі Поттер, герой однойменної серії книжок та фільмів, задача визначення класу, до якого належить даний текст значно ускладняється, на відміну від звичайних новинних текстів, в яких фігурували б ці дані в якості нагадування читачам. Зважаючи на вищесказане, можна зробити висновок, що в

коротких новинних текстах, які в більшості складаються з заголовка та анотації поширене використання власних назв. Власна назва є іменником, який в своєму первинному застосуванні відноситься до унікального об’єкта або явища, такого як Лондон, Юпітер, Сара або Майкрософт, що відрізняється від звичайного іменника, який зазвичай відноситься до класу сутностей (місто, планета, людина, корпорація), або не унікальних прикладів певного класу (міста, іншої планети, цих осіб, нашої корпорації). Врахування власних назв є дуже важливим при класифікації коротких новинних текстів, адже їх часка з-поміж усього тексту (заголовка та анотації) може бути значною. Наприклад, у вже розглянутому реченні власна назва «Деніел Редкліфф» складає 12% від усього тексту та несе у собі найбільше смислове навантаження. Отже, власні назви при класифікації коротких новинних текстів мають значний вплив на її результат, за рахунок високої смислової навантаженності та обмеженому контексту ситуації, що описується, в короткому новинному повідомленні.

Крім того, у соціальних мережах – основному джерелі коротких новинних текстів, відсутній механізм категоризації, такий, який є на сайтах мас-медіа. І навіть автор статті не може віднести її до певної рубрики, тобто відсутня можливість використання мануальної класифікації, саме тому автоматизована класифікація коротких новинних текстів є надзвичайно затребуваною.

Класифікація новин може проводитися критеріями:

- за характером і складом суб’єктів, беруть участь в новині, спрямованістю та масштабом розповсюдження:
  - по соціально-структурному контуру:
    - соцієтальні (на рівні суспільства в цілому);
    - внутрішньо між групові;
    - міжособистісні.
  - за територіальним охопленням

        - глобальні;
        - загальнонаціональні;
        - локальні.
- Зміст і функції переданої інформації:
    - по сфері функціонування:
        - політичні;
        - економічні;
        - освітні;
        - тощо.
    - за тематикою:
        - універсальні;
        - мультитематичні;
        - спеціалізовані.

Автор зосередився на класифікації новин за критерії змісту за сфері функціонування у магістерському дослідженні. Зважаючи на відсутність жорсткого визначеної системи класів, за яким має проводитися дана класифікація було виділено чотири основних класи, на які можна розбити всю сукупність новинних текстів: політика, наука, спорт та розваги. Звісно дану класифікацію можна розширювати та будувати різні ієрархічні моделі для класифікації. Наприклад, розбити клас спорт на підкласи футбол, баскетбол, тощо. Збільшення кількості класів може призвести до зниження точності роботи класифікації, особливо для коротких новинних текстів. Зважаючи на описану вище специфіку коротких новинних текстів, дані тексти характеризуються обмеженню довжину повідомлення, а також нереалізованість, загальними фразами, що інколи робить неможливою ієрархічну класифікацію не знаючи контексту. Наприклад, наступну новину не можливо віднести до підкласу футбол, не маючи додаткових деталей: «Кріштіану Роналду став рекордсменом збірної Португалії за кількістю зіграних матчів». Саме тому використання ієрархічної

класифікації для коротких новинних детальніше не є доцільним, адже може негативно вплинути на точність отриманих результатів.

У класифікації коротких новинних текстів існує ще одна особливість. Інколи новини не стосуються лише одного аспекту людської діяльності. Наприклад, новина «Несподівано: Кріштіану Роналду все ще сумує за Іриною Шейк» може бути віднесена людиною, що здійснювала мануальну класифікацію, одразу до обох класів, а саме спорт та розваги. Кріштіану Роналду – всесвітньо відомий футболіст та безпосередньо пов'язаний зі спортом. Ірина Шейк – модель, представниця шоу бізнесу і може бути віднесена до рубрики розваги. Іншим прикладом може бути поєднання політики та розваг, а саме новина «Барак Обама та його дружина Мішель відвідали презентацію в Національній портретній галереї у Вашингтоні», яка також може бути віднесена до двох класів. Саме тому для класифікації коротких новинних текстів недостатнім є використання мультикласової класифікації. Найбільш доцільний підходом автор вважає проведення мультиміткової класифікації.

1.3 Опис існуючих методів автоматизованої класифікації

Машинне навчання з вчителем базується на визначенні класу об'єкту на основ вже розміченого набору даних (тренувального набору). Кожен метод цього виду обов'язково має дві дії:

- «Тренування» на спеціально приготованих даних, які мають вигляд «об'єкт-реакція». У випадку класифікації коротких новинних текстів – «текстові дані-новини, що відносяться до різних сфер функціонування». При виборі оптимального набору тренувальних даних можна досягнути точної класифікації об'єктів, які не були надані класифікатору для навчання. Наведемо короткий алгоритм «тренування» [9]:

  1) Спочатку збирається колекція документів, на основі якої

навчається машинний класифікатор.

2) Кожен документ розкладається у вигляді вектору ознак (аспектів), за якими він буде досліджуватися.
3) Вказується правильний тип (тематика) для кожного документа.
4) Проводиться вибір алгоритму класифікації і метод для навчання класифікатора.
5) Отримана модель використовується для визначення сфери функціонування документів нової колекції.

- Визначення приналежності об’єкту до певного класу. Ця дія виконується на основі створеної під час «тренування» функції.

Детальніше пропонується зосередитися на етапі розкладання документу у вигляді вектору ознак (аспектів), за якими він буде досліджуватися. Існує декілька підходів для виділення суттєвих ознак з тексту, але найбільш популярним є bag-of-words (від англ. «мішок слів») та word2vec (від англ. «слово до вектора») підходи. Саме на них і пропонується зосередитися детальніше.

1.3.1 Bag-of-words

Мішок слів (або Bag-of-Words) це представлення текстів, написаних натуральною мовою, в якій кожен документ або текст виглядає як невпорядкований набір слів без відомостей про зв’язки між ними. Мішок слів спрощує представлення тексту, що використовується при обробці природних мов та пошуку інформації. У даному представленні текст (наприклад, речення або документ) представляється як сумка (мультисет) слів, ігноруючи граматику і навіть порядок слів, але зберігаючи множинність.

Модель «мішка-слів» зазвичай використовується в методах класифікації документів, де (частота) поява кожного слова

використовується як функція для навчання класифікатора [16]. Так текст можна представити у вигляді матриці, кожен рядок в якій відповідає окремому документу або тексту, а кожен стовпець – окремому слову. Осередок на перетині рядка і стовпця містить кількість входжень слова в відповідний документ.

Серед переваг можна виділити:

- Простота реалізації.

Недоліками є:

- Відносно низька точність в порівнянні з більш складними представленнями.

1.3.2 Word2vec

Word2vec – представлення природомовних текстових даних, що базується на дистрибутивної семантиці (обчислення ступеня семантичної близькості між лінгвістичними одиницями на підставі їх розподілу – дистрибуції) і векторному поданні слів. Word2vec у якості вхідних даних приймає великий текстовий корпус (великий малих лінгвістичних даних) в і зіставляє у відповідність кожному слову вектор, у якості вихідних даних віддаються координати слів. Спочатку Word2vec створює словник, «навчаючись» на вхідних текстових даних, а потім обчислює векторне подання слів. Векторне подання ґрунтується на контекстної близькості: слова, що зустрічаються в тексті поруч з однаковими словами (а отже, мають схожий зміст), у векторному поданні матимуть близькі координати векторів-слів. Отримані вектори-слова можуть бути використані для обробки природної мови та машинного навчання.

У word2vec існують два основних алгоритму навчання: CBOW (Continuous Bag of Words або «безперервний мішок зі словами»).

Серед переваг можна виділити:

- Порівняно висока точність.

Недоліками є:

- Значна складність реалізації в порівнянні з попереднім підходом.

У даній магістерській роботі було обрано представлення тексту у вигляді «мішка слів», адже даний підхід значно простіший в реалізації, але при цьому дає достатньо високі результати в точності. Крім того модифікація мішка слів лежить в основі реалізації word2vec.

Обробка природомовних текстів за допомогою машинного навчання набула широкої вживаності для вирішень різних задач комп'ютерної лінгвістики. Наразі існує безліч методів машинного навчання з вчителем та їх класифікації. Найбільш популярними методами машинного навчання з вчителем для обробки природномовних текстів є: наївний баєсівський класифікатор, метод k-найближчих сусідів, метод опорних векторів, випадковий ліс.

Класифікація новинних текстів є складною задачею, що може бути розв'язана за допомогою мультикласовою або мультиміткової класифікації, як було зазначено вище. Тому для класифікації коротких новинних текстів необхідно визначити найбільш оптимальний за параметрами масштабованості на різну кількість класів, відносної простоти реалізації та використання метод автоматизованої класифікації, який також буде досягати отримати високу точність за заданих умов.

Деякі класифікаційні методи дозволяють використовувати більше двох класів, інші – за своєю природою покликані розв'язувати лише задачі бінарної класифікації. Звичайно, такі бінарні методи можуть бути перетворені в мультикласові класифікатори різними стратегіями, проте завдання даного дослідження не зосереджене на модифікації бінарних методів класифікації до методів, що розв'язують задачі мультикласової а тим більше мультиміткової класифікації. Саме тому при розгляді методів

особливу увагу було приділено можливості їх масштабування на кількість класів, що перевищує значення у 2 класи.

1.3.3 Наївний Баєсів класифікатор

Наївний Баєсів класифікатор – це ймовірнісний класифікатор, який в своїй основі використовує теорему Баєса для визначення ймовірності приналежності спостереження (елемента вибірки) до одного з класів C за умови того, що залежні змінні приймають задані значення: $P(C \mid F_1,...,F_n)$, мається на увазі відносяться до одного з попередньо визначених класів. Така особливість називається наївним припущенням незалежності змінних.

Спрощено принцип роботи даного класифікатора можна описати наступним чином: якщо вхідний об’єкт (в нашому випадку текст) на основі навчальної вибірки можна однозначно віднести до конкретного класу, то результатом класифікації буде ймовірність приналежності до цього класу, яка дорівнює 1; у випадку, якщо досліджуваний об’єкт з різною ймовірністю може належати до різних класів, то результатом класифікації буде вектор, компоненти якого є ймовірностями приналежності до того чи іншого класу.

Починаючи з середини минулого сторіччя наївний Баєсів класифікатор активно досліджувався і досить швидко здобув широку популярність, ставши найбільш використовуваним методом серед інших методів класифікації природомовних текстів. Варто виділить той факт, що незважаючи на свій наївний вигляд, наївний Баєсів класифікатор може надавати точність на рівні сучасних більш складних алгоритмів, крім часто даний класифікатор працює набагато краще в порівнянні з іншими методами, за умови правильного початкового набору даних для навчання. Крім того значною перевагою наївного Баєсів класифікатора є малий обсяг даних для навчання, необхідних для оцінки параметрів, що є необхідними при проведенні класифікації.

Наївний Баєсів класифікатор об’єднує модель з правилом рішення. Одне загальне правило має вибрати найбільш ймовірну гіпотезу. Таке правило відоме як апостеріорне правило прийняття рішення (MAP). Відповідний класифікатор – це функція $classify$, яка реалізується в такий спосіб:

$$classify(f_1,..., f_n) = \arg\max_c p(C = c)\prod_{i=1}^{n} p(F_i = f_i \mid C = c)$$

Серед переваг можна виділити:

- Здатність до навчання на невеликих тренувальних наборів без втрати точності.
- Здатність до інкрементного навчання. Під інкрементним навчання мається наувазі наступне: кожен новий документ, що отримує класифікатор може оновити навчальну базу. Така особливість зумовлена на тому факті, що навчання баєсового класифікатора відбувається ітеративна.
- Висока швидкість роботи.
- Здатність до масштабованість.
- Проста інтерпретація моделей.
- Простота програмної реалізації.
- Відносно малі вимоги до оперативної пам’яті.
- Можливість як бінарної так і мультикласової класифікації.
- Здатність до мультиміткової класифікації.

Недоліками є:

- Для текстових даних припущення незалежності не справджується. Хоча на практиці це не дуже суттєво впливає на результуючу якість класифікації природомовних текстових даних.

1.3.4 Метод найближчих сусідів

Метод найближчих сусідів (англ. nearest neighbors algorithm, NN) – непараметричний, метричний алгоритм для автоматизованої класифікації об’єктів. В основі такого класифікації об’єкта у рамках простору властивостей лежать відстані, пораховані до усіх інших об’єктів. В результаті, обирається та група об’єктів, до яких відстань є найменшою, і вони виділяються в окремий клас. Отже, основним принципом методу найближчих сусідів є те, що об’єкт присвоюється той класу, який є найбільш поширеним серед сусідів даного елемента. Це твердження базується на так гіпотезі компактності: «в просторі об’єктів все близькі об’єкти повинні належати до одного класу, а все різні об’єкти відповідно повинні перебувати в різних класах».

Сусіди беруться виходячи з безлічі об’єктів, класи яких вже відомі, і, виходячи з ключового для даного методу значення k вираховується, який клас найбільш численний серед них. Кожен об’єкт має кінцеве кількість атрибутів (розмірностей).

Передбачається, що існує певний набір об’єктів з уже наявною класифікацією.

Під час роботи алгоритму зазвичай документи подаються у вигляді векторної моделі семантики, а для оцінки відстані зазвичай використовують евклідову відстань. Тобто відстань між двома точками рахується як евклідова відстань (інколи її ще називають евклідовою метрикою), відповідно за формулою традиційної відстані між двома точками для скінчено вимірного дійсного векторного простору E із скалярним добутком (Евклідового простору) [17]. Хоча в реалізаціях іноді теж зустрічаються: манхетенська відстань, косинусна міра, критерій кореляції Пірсона та інші.

Метод найближчих сусідів зазвичай поділяється на:

- Метод найближчого сусіда – найпростіша версія алгоритму.

Об’єкт відноситься до того ж класу, що і його найближчий сусід.

- Метод k найближчих сусідів – найпоширеніша версія для класифікації з двома класами. Об’єкт відносять до того ж класу, що і більшість його сусідів. В задачах з парної кількістю класів рекомендується брати непарне k, щоб запобігти появі невизначеності.
- Метод k зважених найближчих сусідів. В задачах з великою кількістю класів часто все одно можуть виникнути невизначеності. Саме тому рекомендується кожному i-му сусіду призначати вагу. Зазвичай вага зменшується зі зростанням рангу i сусіда. Об’єкт відноситься до класу, який набрав найбільшу сумарну вагу серед k найближчих сусідів.

Серед переваг можна виділити:

- Простота реалізації.
- Стійкість до аномальних викидів. Оскільки ймовірність потрапляння такого об’єкту в число k найближчих мала. Навіть якщо це відбулося то вплив на голосування, а особливо на зважене голосування буде дуже не значний.
- Висока швидкість роботи.
- Проста інтерпретація моделей.

Недоліками є:

- Потрібно дуже точно підбирати вибірку. Оскільки на нерепрезентативній вибірці результати роботи алгоритму будуть незадовільні.
- Потребує велику кількість операцій для класифікації вибірки. Що особливо відчутно при наявності великої навчальної вибірки.
- Не дуже висока точність в більшості задач.

1.3.5 Метод опорних векторів (SVM)

Метод опорних векторів – метод класифікації, що належить до групи граничних методів, тобто принцип роботи даного класифікатора базується на визначенні, до якого з класів відноситься об'єкт класифікації, за допомогою меж просторів; це набір схожих алгоритмів виду «навчання із вчителем». Ці алгоритми зазвичай використовуються для задач класифікації та регресійного аналізу. Метод належить до розряду лінійних класифікаторів. Особливою властивістю методу опорних векторів є безперервне зменшення емпіричної помилки класифікації та збільшення проміжку. Тому цей метод також відомий як метод класифікатора з максимальним проміжком. Основна ідея методу опорних векторів – перевід вихідних векторів у простір більш високої розмірності та пошук роздільної гіперплощини з максимальним проміжком у цьому просторі. Дві паралельні гіперплощини будуються по обидва боки гіперплощини, що розділяє наші класи. Роздільною гіперплощиною буде та, що максимізує відстань до двох паралельних гіперплощин. Алгоритм працює у припущенні, що чим більша різниця або відстань між цими паралельними гіперплощинами, тим меншою буде середня помилка класифікатора [18]. Тобто, клас об'єкта визначається потраплянням досліджуваного об'єкта до однієї з площин, визначеної класи. Опорними векторами вважаються об'єкти множини, що лежать на межах визначених просторів. Процес класифікації вважається вдалим, якщо простір між межами є порожнім.

Серед переваг можна виділити:

- Найшвидший метод знаходження вирішальних функцій.
- Не потребує великих обсягів пам'яті.
- Висока точність в більшості задач.
- Ефективно працює при великій кількості ознак.

Недоліками є:

- Прямо використати можна тільки до класифікації на 2 класи.
- Складна інтерпретація моделей.
- Повільне навчання.
- Чутливий до шумів.
- Не можна отримати ймовірність відношення до класу на пряму.

1.3.6 Випадковий ліс

Random forest (англ. випадковий ліс) – алгоритм машинного навчання, запропонований Лео Брейманом [19] і Адель Катлер, що полягає у використанні комітету (ансамблю) вирішальних дерев. Алгоритм поєднує в собі дві основні ідеї: метод беггінга Брейман і метод випадкових підпросторів, запропонований Tin Kam Ho. Алгоритм застосовується для задач класифікації, регресії і кластеризації.

Класифікація відбувається шляхом голосування. Об’єкт відноситься до того класу, до якого його віднесли більшість дерев.

Перевагами є:

- Ефективна робота на даних з великою кількістю ознак та класів.
- Масштабованість.
- Надає можливість оцінювати значущість окремих ознак моделі [19].
- Здатність до розпаралелення.

Недоліками є:

- Великій розмір отримуваних моделей, а значить потреба у великій кількості оперативної пам’яті.
- Схильність до перенавчання.
- Чутливість до шумів.

1.4 Висновки

Аналіз існуючих підходів до автоматизованої класифікації природомовних текстів, в тому числі новинних, показав, що на сьогоднішній день все ще активно використовується мануальна (інтелектуальна) класифікація, яка показує свою життєспроможність для вирішення задачі класифікації новинних текстів на сторінках засобів масової інформації в Інтернеті. Дослідження особливостей коротких новинних текстів, як популярного напрямку розвитку сучасної Інтернет-журналістики, показав складності, що існують у мануальній та автоматизованій класифікації. Основними, найбільш істотними недоліками існуючих методів є те, що вони не здатні враховувати підтекст, що несуть у собі імена відомих людей, назви міст, спортивних клубів тощо. Також більшість підходів орієнтуються лише на визначенні одного класу новини та не проводять мультиміткову класифікацію.

Досліджено важливість врахування власних назв при класифікації коротких новинних текстів, за рахунок високої смислової наванженості, що власні назви несуть у коротких новинних текстах.

Зважаючи на особливості коротких новинних текстів, для якісної класифікації враховувати їх особливості, а саме: лаконічність, наявність власних назв та схильність до віднесення до декількох тематик. Саме тому було обрано метод представлення вхідного тексту у вигляді мішка слів. Процес класифікації було вирішено проводити за допомогою наївного Баєсового класифікатору, який дозволяє здійснювати багатокласову та мультиміткову класифікацію, не вимагає великого вхідного набору для навчання, простий в реалізації та показує високі результати точності в порівнянні з більш складними методами класифікації природомовних текстових даних, зокрема коротких новинних текстів.

Отже, актуальною є задача розробки методу автоматизованої класифікації коротких новинних текстів, який дозволить покращити якість

рубрикації коротких новинних текстів за рахунок врахування їх особливостей.

2 РОЗРОБЛЕНИЙ МЕТОД КЛАСИФІКАЦІЇ КОРОТКИХ НОВИННИХ ТЕКСТІВ

2.1 Особливості коротких новинних текстів

Як вже зазначалося вище, користувачам, що підписані на сторінки соціальних медіа, стає просто неможливо відстежувати всю сукупність новинних повідомлень. При цьому варто зазначити, що більшість читачів не зацікавлені у всіх темах, а мають

Для журналістів найбільшою цінністю є соціальна та інформаційна функції Інтернету. Розуміючи переваги глобальної мережі, медіа кинулися її освоювати. Постійні розмови про смерть друку через телебачення і радіо зараз трохи змінили ракурс. Тепер саме Інтернет називають причиною загибелі традиційних ЗМІ. Але поки ми спостерігаємо іншу ситуацію: завдяки глобальній мережі, ці самі традиційні медіа розширюють свою аудиторію, розвиваються і знаходять нові формати роботи. Так, наприклад, Інтернет надав читачам ЗМІ нові можливості: лишати відгук на отриману інформацію та створювати власний контент, що може бути згодом використаний у ЗМІ. У той же час виробники новин теж оперативно реагують на зауваження читачів. Як зазначається в книзі «Журналістика та конвергенція»: «В мультимедійних ЗМІ збільшуються можливості … індивідуального налаштування та індивідуального втручання користувача в зміст ЗМІ (можливість залишити коментар на сайті, написати відгук на статтю, проголосувати за матеріал, відправити посилання одному і т.д.)» [20].

Не тільки користувачі зацікавлені в онлайн-ЗМІ, багато медіа-менеджерів говорять про вигоду Інтернет-комунікацій і бачать за нею майбутнє ЗМІ. За даними на 2008 рік, 44% медіаменеджерів вибрали Інтернет в якості медіа платформи [21].

Однак в останні декілька років великої популярності набуло нове медійне явище - соціальні мережі. Наразі вони відіграють значну роль у

житті більшості користувачів мережі Інтернет. Слід відмітити, що сучасні соціальні мережі – це не лише місце для спілкування: за останні п'ять-сім років їх функції значно розширилися. Все частіше фахівці говорять про інтеграцію медіа і соціальних мереж та перенесення функцій ЗМІ на бік останніх. Соціальні мережі поступово стають джерелом різноманітної інформації, і ця функція є незамінною як для користувачів соціальних мереж так і для ЗМІ. Крім того, читач став ближче до журналіста: у коментарях до постів можна спостерігати миттєву реакцію людей на ту чи іншу інформацію.

Сьогодні ми відзначаємо початок взаємопроникнення між світом новин та соціальними мережами. Наочний приклад – зростаюча передача новин по каналах Facebook. Більшість медійних компаній пропонують своїм журналістам використовувати соціальні мережі такі як, наприклад, Twitter [22]. Якщо переглядати сайти ЗМІ, то практично на всіх можна знайти іконки Facebook, ВКонтакте і Twitter. Це новий крок до зближення видань з читачами.

На конференції News Online Association восени 2015 року було наведені такі дані: близько двох третин користувачів Facebook і Twitter отримують новини з соціальних мереж. Це більше, ніж в 2013 році, коли відсотки були 47% для Facebook і 52% для Twitter [23].

Приблизно такі ж результати ми знайдемо в звіті WorldPressTrends2015: «0,8 мільярдів – 42% всіх Інтернет-користувачів читають новини в форматі digital» (цифровий формат) [24].

Отже, зважаючи на вищесказане, можна зробити висновок, що невід’ємною частиною більшості новинних видань стало представлення новинних статей на сторінках онлайн медіа-ресурсів, що в свою чергу тісно інтегровані у великі соціальні мережі, такі як Facebook і Twitter.

2.2 Характерні риси коротких новинних текстів в соціальних мережах

У 1991 році вийшла книга «Eyes on the News» М.Гарсіа і П.С.Адама з дослідженням того, як люди читають друковану пресу. У книзі наведені наступні висновки: «в першу чергу читачі звертають увагу на кольорову фотографію, потім на заголовок, підпис під зображенням, бриф (історії скорочуються від одного до трьох параграфів) і на ряд інших графічних пристроїв, званих точками входу – місце, з якого читач починає читати історію» [25]. У дослідженні також йдеться про те, що більшість людей тільки побіжно переглядають газети і вкрай рідко дочитують тексти до кінця. Вони порахували, що середній читач переглядає близько 25% однієї історії в газеті, але уважно прочитує трохи менше [26].

Однак Інтернет вніс істотні поправки до форми подання, формату, принципів відбору новин та їх читання в мережі.

Проведене в 2007 році дослідження «EyeTrack07» дозволило краще зрозуміти, як саме люди читають новинні матеріали в Інтернеті.

Дослідники С. Куїнн і П. С. Адам отримали наступні дані:

1. Люди вибирають для читання те, що хочуть прочитати.
2. Люди читають онлайн набагато більший відсоток тексту, ніж у друкованій пресі.
3. У середньому онлайн читачі прочитали 77% тексту, який вони вибрали для читання.
4. Майже дві третини онлайн читачів дочитують до кінця обраний ними текст.
5. Близько половини онлайн читачів «сканували» сторінки по діагоналі, в той час як інша половина була послідовною в читанні.

6. Серед онлайн-читачів була невелика різниця в кількості прочитаного тексту між «скануючими» і методичними читачами.
7. Такі речі, як питання-відповідь, хронологія, блоки фактів або короткі списки допомагають читачам зрозуміти і запам’ятати, що вони читали. [27].

В першому розділі, під час аналізу особливостей класифікації коротких новинних текстів, було відзначено, що суттєву роль класифікації коротких новинних текстів відіграє врахування контексту, що несуть у собі власні назви. В комп’ютерній лінгвістиці власні назви отримали визначення іменованих сутностей. Іменовані сутності – це об’єкти реального світу, такі як особи, місця розташування, організації, продукти тощо, які можуть бути позначені власним ім’ям. Іменована сутність не обмежена жодними рамками і може вказувати як на абстрактний об’єкт так і на об’єкт, що фізично існує. Прикладами іменованих сутностей є «Барак Обама», «Нью-Йорк», «Volkswagen Golf» і тому подібне. Отже, іменована сутність – це все, що можна назвати власною назвою. Іменовані сутності можна просто розглядати як примірники об'єктів (наприклад, Нью-Йорк – це примірник міста). Як було показано в прикладі, вага іменованих сутностей при класифікації новинних текстів звичайного обсягу є значною, проте при класифікації коротких новинних текстів, де кількість символів обмежена, іменовані сутності можуть стати ключовим фактором, що допоможе покращити якість класифікації. Наприклад, вживання іменованої сутності Барак Обама може свідчити про приналежність новини до класу «політика» або «світське життя».

Отже, іменовані сутності можуть значно покращити якість класифікації коротких новинних текстів, а тому при наявності іменованих сутностей в тексті їх можна використовувати, як одну з ознак для класифікації. Так, використання іменованих сутностей не є новим у сфері

комп’ютерної лінгвістики, проте класифікація коротких новинних текстів із застосуванням аналізу іменованих сутностей досі не досліджувалося.

При аналізі особливостей подання новинних текстів в рамках онлайн платформ, а саме соціальних мереж Facebook та Twitter, будемо досліджувати такі їх функціональні характеристики:

– довжина постів;
– можливість прикріплення до посту файлів;
– можливість прикріплення до посту посилань;
– алгоритм побудови новинної стрічки (особливості появи постів у стрічці новин);
– інструменти оцінки реакції на дописи інших користувачів – варіанти оцінювання чужих повідомлень.

2.2.1 Тексти в соціальній мережі Facebook

Серед особливостей цієї платформи можна виділити те, що пости в Facebook мають довжину близько 240 символів [28]. Тексти, довжина яких перевищує цей обсяг, згортаються під посилання «Ще». При цьому максимальна довжина повідомлень на «стіні» користувача становить 63206 символів.

У мережі Facebook є можливість прикріплення до постів файлів різноманітних форматів (зображень, відео та аудіо матеріалів), а також посилань на інші ресурси. У Facebook з 2016 року введена розширена шкала реакцій на пости, окрім стандартного вподобання (like), було введено 5 нових реакцій, що висловлюють наступні емоції:

- кохання,
- сміх,
- подив,
- смуток,
- обурення.

Оскільки ці метрики не можуть жодним чином вплинути на класифікацію тексту (тільки при комплексній класифікації посту), вони не будуть використовуватися при класифікації новинних текстів.

У Facebook є можливість алгоритмічного та хронологічного підходів до побудови новинної стрічки. За замовченням увімкнено алгоритмічний підхід, який обирає найбільш релевантні за версією Facebook пости, але не завжди задовольняє потреби користувачів.

2.2.2 Тексти в соціальній мережі Twitter

Особливістю Twitter є довжина повідомлення – всього 140 символів для будь-якої мови світу. «За змістом Twitter – це віртуальна майданчик для спілкування: миттєвого поширення і отримання повідомлень довжиною не більше 140 символів. А за формою – спілкування в SMS-форматі, ідею якого засновники компанії перенести з телефонів в Інтернет і назад» [29].

З символів складається повідомлення – основна комунікаційна одиниця Twitter. Англійською воно називається tweet.

Популярність Twitter призвела до того, що короткі повідомлення стали новою формою поширення інформації. Виявилося, що 140 символів можуть передати повідомлення будь-якого характеру: рекламу, політичні погляди, настрій і думки.

Ще одна нещодавно змінена особливість Twitter – це зміна хронологічного порядку відображення повідомлень на їх сортування за релевантністю на основі закритих алгоритмів Twitte [29], що зайвий раз свідчить про необхідність більш розумної організації стрічки новин.

Зазвичай в повідомленнях Twitter медіа ресурси публікують посилання на свої повні статті на відповідному сайті. Окрім посилання, в твіт додається анотація статті, яка зазвичай співзвучна з заголовком цієї статті. В заголовках можуть бути наявними іменовані сутності, які можна

врахувати під час класифікації, адже короткий текст повідомлення може не нести в собі ніяких особливих рис, що значно ускладнює його обробку.

2.3 Метод класифікації коротких новинних текстів з використанням іменованих сутностей

Зважаючи на визначені вище особливості коротких новинних текстів в соціальних мережах, автори статті пропонують впровадити для класифікації такого роду текстів два підходи: перший, який базується на аналізі загального тексту новини або її анотації; інший – використовує виключно наявні в новині іменовані сутності як ознаку класифікації, якщо таких немає – класифікація відбувається лише за рахунок класифікатора, який базується на аналізі загального тексту новини.

Для реалізації обох підходів пропонується використати наївний баєсівський класифікатор через такі переваги як:

- висока швидкість роботи;
- потреба в невеликій кількості даних для навчання;
- невимогливість до обсягу використовуваної пам’яті.

Для класифікації новин, яка базується на аналізі загального тексту новини або її анотації, пропонується використати стандартний підхід до оброблення природомовних текстових даних:

- використання сторонніх бібліотек для передоброблення тексту (лематизації, стемізації, тощо).
- відкидання сполучників та інших службових слів, що не несуть додаткової інформації для статистичного класифікатору.
- навчання на вже розміченій вибірці.
- використання для класифікації нових даних.

Класифікатор, що базується на врахуванні іменованих сутностей, пропонується використовувати наступним чином:

– ознакою для навчання обрати іменовані сутності з тексту;
– для визначення іменованих сутностей використовувати метод, що базується на нейронних мережах;
– провести навчання на вже розміченій вибірці, що містить іменовані сутності в кожній одиниці вибірки;
– проводити класифікацію лише за умови детекції іменованої сутності в тексті.

Даний класифікатор може бути використаний лише на текстах, в яких наявні іменовані сутності, бо в іншому випадку його результати можуть негативно вплинути на отримуваний результат.

Пропонується виконувати дії щодо класифікації вхідного тексту в такому порядку:

1. Передоброблення тексту (лематизація, стемізація, тощо).
2. Пошук іменованих сутностей. Для вирішення цієї задачі можна використати, наприклад, нейронну мережу або якийсь з інших існуючих підходів.
3. При наявності іменованих сутностей – класифікація тексту з використанням класифікатору, що використовує іменовані сутності, як ознаки.
4. Класифікація тексту з використанням класифікатору, який базується на загальному тексті.
5. Якщо є 2 результати – об’єднання результатів класифікаторів за одною зі стратегій. Наприклад, найпростіший варіант – використання середнього арифметичного оцінок обох класифікаторів – обчислення середнього арифметичного для значень оцінок, що визначив кожен з класифікаторів.

2.4 Висновки

У даному розділі розглянуто метод автоматизованої класифікації коротких новинних текстів. Виконано аналіз існуючих тенденцій у галузі онлайн журналістики, досліджено особливості коротких новинних текстів та проведено порівняльний аналіз коротких новинних текстів у різних соціальних мережах.

В рамках дослідження, описаного в даному розділі, були проаналізовані визначенні особливості з точки зору їх врахування при автоматизованій класифікації коротких новинних текстів, на основі чого були висунуті відповідні гіпотези, а саме те, що використання іменованих сутностей при класифікації коротких новинних текстів може покращити результи класифікації.

На основі даної гіпотези було запропоновано власний метод автоматизованої класифікації коротких новинних текстів, що базується на використанні іменованих сутностей. На основі проведеного дослідження було викладено теоретичний матеріал, що описує запропонований метод.

3 ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ ДЛЯ АВТОМАТИЗОВАНОЇ КЛАСИФІКАЦІЇ КОРОТКИХ НОВИННИХ ТЕКСТІВ

3.1 Обґрунтування вибору засобів реалізації

Для створення бібліотеки класів необхідно визначитися з мовою програмування та середовищем розробки програмного забезпечення. В даному підрозділі буде розглянуто декілька найпопулярніших мов програмування.

3.1.1 Вибір мови програмування

П’ятірка лідерів в галузі розробки програмного забезпечення серед мов програмування є незмінною впродовж останніх років [30]. Найбільш популярними мовами програмування на сьогоднішній день є Java, C, C++, Python та C#. Вибір мови програмування проводився серед наступних мов Java, Python та C#. Мови програмування C та C++ були відкинуті відразу. Мова C, тому що не підтримує концепцію об’єктно орієнтованого програмування, не підтримує побудову програмного забезпечення за допомогою простору імен, не має поняття конструктора чи деструктора. Мова програмування C++ має наступну низку недоліків: важко підтримувати та розширювати у високорівневий програмних застосунках, не підтримує збирання сміття, не є безпечною через наявність вказівників, «дружніх функцій» та глобальних змінних. Саме через те, що при написання програмних засобів мовами C та C++ дуже легко допустити помилок, що спричинять «протікання пам’яті» тощо, дані мови програмування не розглядалися.

Мова програмування C#

C# – це типізована об’єктно-орієнтована мова програмування від компанії Microsoft, мультипарадигмальна мова програмування, яка включає в себе об’єктно-орієнтовну, імперативну, функціональну,

рефлективно-орієнтовну, подійно-орієнтовну, задачно-орієтовну парадигми.

Синтаксис мови багато належить до сімейства C-подібних мов.

Мова програмування C# підтримує [31]:

- Поліморфізм.
- Перевантаження операторів (в тому числі операторів явного і неявного приведення типу).
- Делегати.
- Атрибути.
- Події.
- Властивості.
- Узагальнені типи і методи.
- Ітератори.
- Анонімні функції з підтримкою замикань.
- LINQ.
- Виключені ситуації.
- Коментарі в форматі XML.

Важливим фактом є те, що мова C# використовує автоматичне керування пам’яттю, що усуває більшість ризиків пов’язаних з мовами C та C++.

У версії мови C# 7 додалося безліч зручних функціональних можливостей, наприклад, покращені Out змінні, доданий паттерн порівняння зі зразком (від англ. Pattern matching), додано кортежний тип данних – тупли (tuples), можливість деконструкції, локальні фунції тощо. Всі ці нововведення спрощують написання коду та зменшують часові затрати [32].

Віднедавна Microsoft керує розробкою компілятора C# з відкритим вихідним кодом та набором інструментів, що раніше мав кодову назву «Roslyn» [33]. Компілятор, який реалізовано за допомогою керованого

коду (C#), був відкритий, а функціональні можливості представлені у вигляді API. Це дозволяє розробникам власноруч створювати інструменти для рефакторингу та діагностики програмного забезпечення.

В червня 2016 року компанія Microsoft випустила новий феймворк – .NET Core. Даний фрейворк є кросплатформним (може працювати під операційними системати Windows, Mac, Linux) аналогом .NET Framework з відкритим вихідним кодом [34].

Переваги:

- Мова постійно розвивається.
- Можливість компонентного встановлення необхідних засобів розроблення програмного забезпечення (наприклад, лише для розроблення мобільних додатків тощо).
- Компілятор з відкритим кодом.
- Велика база документації з детальним описом класів та методів для кожної версії та великою кількістю прикладів (MSDN).
- Автоматичне керування пам’яттю.
- Велика вбудована бібліотека.
- Можливості зручної роботи з колекціями (LINQ).
- Висока швидкість роботи.
- Зручний інструментарій для розробки багатопоточного програмного забезпечення.
- Можливість створювати інструменти для рефакторингу та діагностики програмного забезпечення.
- Можливість використання елементів функціонального програмування.

Раніше ключовим недоліком було те, що найстабільніша версія була прив’язана до платформи .NET. Завдяки появі .NET Core C# став

кросплатформеною мовою програмування і тепер нічим не поступається кросплатформенним аналогам.

Мова програмування Java

Java – типізована об'єктно-орієнтована мова програмування, від компанії Sun Microsystems, якою нині володіє компанія Oracle.

Синтаксис мови багато належить до сімейства C-подібних мов.

Одним з головних гасел мови є «Напиши один раз – запускай де завгодно» (англ. Write once, run anywhere, WORA). Це означає, що код на Джаві може бути розроблений одного разу і скомпільований в байт-код і потім запущена на довільному пристрої, який має JVM. Але на практиці реалізації JVM на різних ОС мають деякі відмінності і тому часто треба перевіряти роботи розробленого ПЗ на всіх поєднаннях JVM і ОС, які потрібно підтримувати.

Java є суворо типізованою мовою, кожна змінна та вираз має тип, відомий на етапі компіляції.

Переваги:

- Кросплатформеність.
- Великої вбудована бібліотека.
- Мова постійно розвивається.
- Автоматизована робота з пам'яттю.
- Велика база документації з детальним описом класів та методів для кожної версії та великою кількістю прикладів.
- Безпечна мова.

Недоліки:

- Мова є відносно повільною.
- При використанні коду на декількох платформах все одно необхідно детально тестувати та підлаштовувати його під кожну зв'язку ОС та JVM.

- Використовує велику кількість оперативної пам’яті.

В порівнянні з тою ж таки мовою програмування C# Java хоч і розвивається, але робить це досить повільно. Наприклад, повноцінна підтримка лямбда-виразів з’явилася в Java лише в 2014 році, в той час, як C# підтримував їх з 2007 року.

Мова програмування Python

Python – високорівнева мова програмування загального призначення, орієнтований на підвищення продуктивності розробника та читання коду. Синтаксис ядра Python є мінімалістичним. У той же час стандартна бібліотека даної мови програмування включає великий обсяг корисних функцій.

Python підтримує кілька парадигм програмування, а саме наступні парадигми:

- Структурне.
- Об’єктно-орієнтоване.
- Функціональне.
- Імперативне.
- Аспектно-орієнтоване.

Основні архітектурні риси:

- Динамічна типізація.
- Автоматичне керування пам’яттю.
- Повна інтроспекція.
- Механізм обробки виключень.
- Підтримка багатопоточних обчислень і зручні високорівневі структури даних.

Python підтримує динамічну типізацію, тобто тип змінної визначається лише під час виконання. Тому замість «присвоювання

значення змінної» краще говорити про «зв’язуванні значення з деяким ім’ям».

CPython є основною, але не єдиною реалізацією мови програмування Python. Існують також інші реалізації:

- Jython – реалізація Python, що використовує JVM в якості середовища виконання. Дозволяє прозоро використовувати Java-бібліотеки [35].
- IronPython – Python для .NET Framework і Mono. Компілює Python програми в MSIL, таким чином надаючи повну інтеграцію з .NET-системою [36].
- Tinypy [37] – мінімалістична версія Python. Частина можливостей CPython не реалізована.

Python з пакетами NumPy, SciPy і Matplotlib активно використовується як універсальне середовище для наукових розрахунків в якості заміни поширеним спеціалізованим комерційним пакетам Matlab, IDL і іншим.

Переваги:

- Активно підтримується та постійно розвивається.
- Наявність великої стандартної бібліотеки.
- Наявність великої спільноти розробників.
- Кросплатформеність.
- Можливість використання елементів функціонально програмування.
- Наявність інтерактивного режиму.
- Наявність великої бази офіційної документації.

Недоліки:

- Відносно повільна швидкодія.

- Не сумісність 2 та 3 версії.
- Складнощі з використанням Юнікоду в Пітоні Python 2.
- Складна та не завжди надійна реалізація багатопоточності.
- Потребує багато оперативної пам’яті [38].

Отже, зважаючи на розглянуті вище мови програмування, ключовими недоліками Python та Java є відносно повільна швидкодія та високе використання оперативної пам’яті. Саме тому для реалізації програмного забезпечення було обрано мову C#.

3.1.2 Вибір середовища розробки

Зважаючи на те, що для реалізації запропонованого методу було обрано мову C#, серед середовищ розроблення програмного забезпечення розглядаються наступні:

Середовище розроблення Microsoft Visual Studio

Microsoft Visual Studio – це інтегроване середовище розробки (IDE – integrated development environment) від розробника мови C# – компанії Microsoft. Microsoft Visual Studio або просто Visual Studio використовується для розробки комп’ютерних програм, а також веб-сайтів, веб-програм, веб-сервісів та мобільних додатків. Visual Studio використовує різноманітні платформи для розробки програмного забезпечення від Microsoft, а також надає можливість компонентно встановлювати засоби для розроблення. Дане середовище розроблення програмного забезпечення може створювати як некерований – власний код, так і керований код.

Visual Studio включає в себе редактор коду, що підтримує IntelliSense та рефакторинг коду. Visual Studio має велику кількість вбудованих інструментів, наприклад, дизайнер класів та дизайнер схеми баз даних, а також приймає плагіни, які підвищують функціональність практично на

всіх рівнях, включаючи додавання підтримки для систем контролю версій (наприклад, Subversion та Git).

Visual Studio підтримує 36 різних мов програмування.

Найбільшою перевагою даного середовища те, що базова версія Visual Studio Community Edition є доступною безкоштовно.

Останні оновлення запровадили оптимізацію роботи, що спрощує розроблення великих проектів.

Переваги:

- Постійно оновлюється.
- Можна додавати плагіни.
- Має підтримку IntelliSense.
- Найкраща підтримка можливостей середовища.

Недоліки:

- Для комфортної розробки вимагає багато оперативної пам’яті та швидкий диск.

Середовище розроблення SharpDevelop

SharpDevelop (також стилізований #develop) – це інтегроване середовище розробки з вільним і відкритим вихідним кодом для .NET Framework, Mono, Gtk# і Glade#. SharpDevelop підтримує 6 мов програмування.

SharpDevelop був розроблений в якості вільної і «легкої» альтернативи Microsoft Visual Studio, і містить еквівалентні функціональні можливості для майже всіх істотних функція Visual Studio Express, включаючи функціональні можливості для управління проектами, редагування коду, додатки компіляції та відлагодження. Для того, щоб дозволити легку міграцію проекту, SharpDevelop працює файлами проектів Visual Studio (*.sln), а також здатний компілювати програми для .NET Framework версії 2.0, 3.0, 3.5, 4.0 і .NET Framework 2.0 і 3.5 Compact.

Хоча SharpDevelop не такий поширений, як в Visual Studio, SharpDevelop є досить популярним і станом на 2013 рік було завантажено принаймні 8 мільйонів разів по всьому світу.

Переваги:

- Низькі вимоги до апаратного забезпечення.
- Кросплатформеність.
- Можна додавати плагіни.

Недоліки:

- Відстає в підтримці останніх можливостей мови.
- Має тільки 32 бітну версію.
- Урізані функціональні можливості зневаджувача.

Середовище розроблення JetBrains Rider

JetBrains Rider – кроссплатформенная інтегроване середовище розробки програмного забезпечення для платформи .NET від компанії JetBrains, реліз якого відбувся в 2017 році. Підтримує 3 мови програмування.

В основі JetBrains лежить інший продукт JetBrains – ReSharper. Середовище підтримує платформи .NET Framework, .NET Core і Mono. Працює на операційних системах Windows, MacOS, Linux.

Переваги:

- «Розумна» навігація.
- Кросплатформеність.
- «Розумні» функції редагування коду.
- Наявна перевірка на наявність помилок (з підказками).
- Спрощує написання і рефакторінг коду.

Недоліки:

- Урізані функціональні можливості зневаджувача.

- Неможливе повноцінне налагодження асинхронного коду.
- «Гальмує» на великих файлах.
- Відсутня безкоштовні версії.

Зважаючи на те, що SharpDevelop відстає в підтримці останніх можливостей мови, за які і було обрано C#, а JetBrains Rider має лише платну версію, тому середовищем розроблення було обрано Microsoft Visual Studio.

3.2 Архітектурна організація програмного застосунку

Розроблену систему можна розділити на десять модулів, зображених на рис. 3.3.1:

1. Модуль для виділення меж речень – містить інструменти для розбору вхідного тексту на синтаксичні одиниці – речення. На вхід приймає короткий новинний текст. На вихід віддає масив, що складається з речень вхідної новини. Використовується класифікатором, що базується на всьому тексті новини, та класифікатором, що базується на іменованих сутностях.

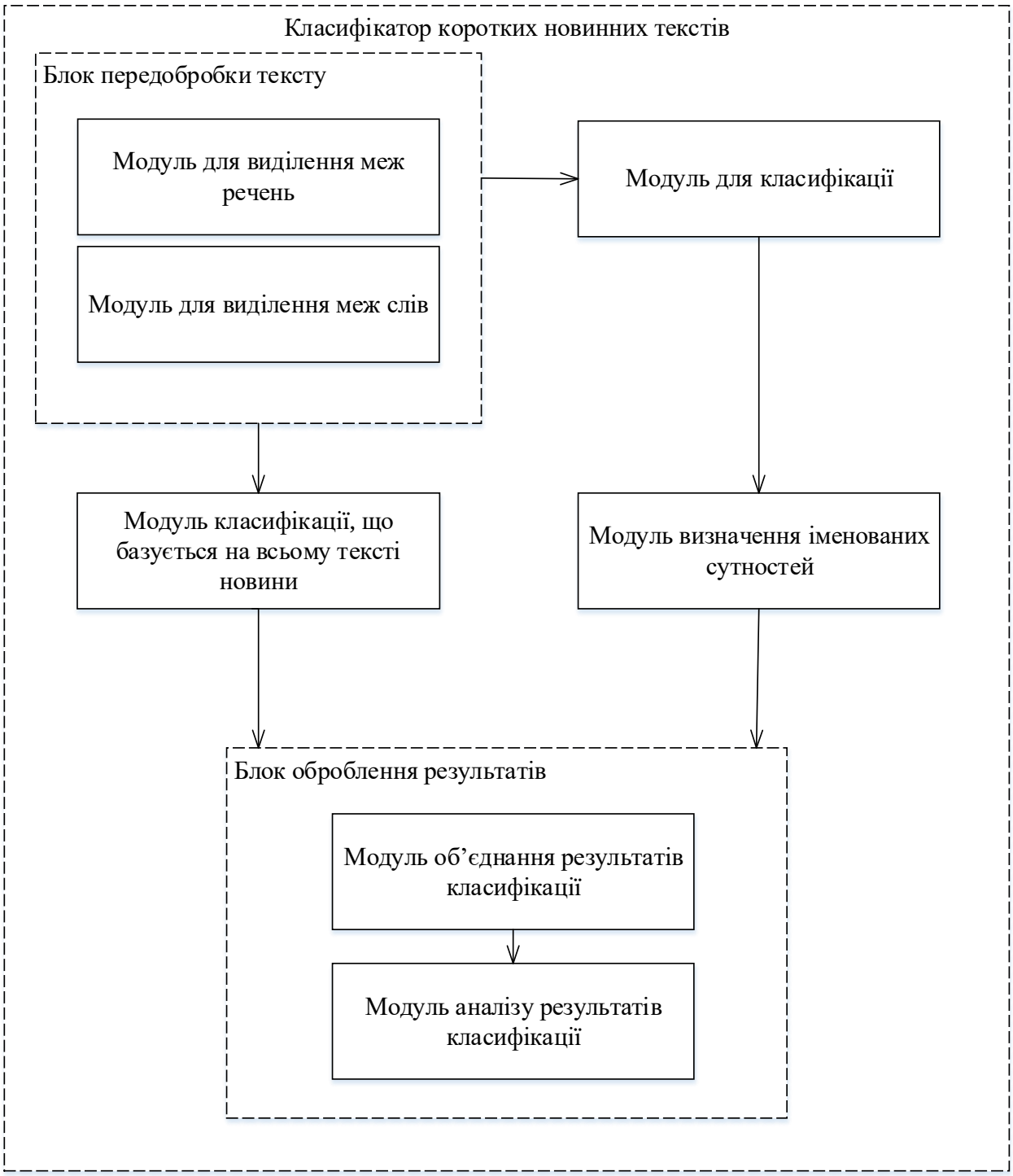

Рис. 3.2.1. Схема модулів програми

2. Модуль для виділення меж слів – містить інструменти для розбору вхідного тексту на лексичні одиниці – слова. На вхід приймає короткий новинний текст (може приймати і результат роботи модуля для виділення меж речень, тобто масив речень). На вихід віддає масив, що складається зі слів вхідної новини.

Використовується класифікатором, що базується на всьому тексті новини, класифікатором, що базується на іменованих сутностях, та модулем визначення іменованих сутностей.

Модуль для виділення меж речень та модуль для виділення меж слів можна об’єднати в блок передобробки тексту, оскільки вони готують вхідний текст до подальшого оброблення, а саме, класифікації коротких новинних текстів.

3. Модуль визначення іменованих сутностей – містить інструменти для вичлеванування з вхідного тексту власних назв – іменованих сутностей. На вхід приймає масив слів. На вихід віддає масив, що складається зі іменованих сутностей. Використовується класифікатором, що базується на іменованих сутностях.
4. Модуль класифікації, що базується на всьому тексті новини – містить інструменти, що класифікує новинний текст за допомогою наївного Баєсового класифікатора. На вхід приймає масив слів. На вихід віддає словник з ймовірностями приналежності новинного тексту до тієї чи іншої категорії. Результати використовуються модулем об’єднання результатів класифікації.
5. Модуль визначення іменованих сутностей – містить інструменти, що класифікує новинний текст використовуючи іменовані сутності як ознаки для класифікації за допомогою наївного Баєсового класифікатора. На вхід приймає масив іменованих сутностей. На вихід віддає словник з ймовірностями приналежності новинного тексту до тієї чи іншої категорії. Результати використовуються модулем об’єднання результатів класифікації.
6. Модуль об’єднання результатів класифікації – містить інструменти, що об’єднують результати роботи n-класифікаторів.

На вхід приймає масив результатів роботи класифікаторів. На вихід віддає об’єднаний словник з ймовірностями приналежності новинного тексту до тієї чи іншої категорії. Результати використовуються в модулі аналізу результатів класифікації.

7. Модуль аналізу результатів класифікації – містить інструменти, що дозволяють прийняти кінцеве рішення щодо класів, до яких відноситься об’єкт. На вхід приймає об’єднаний словник з ймовірностями приналежності новинного тексту до тієї чи іншої категорії. На вихід віддає список міток, які згідно класифікації можна застосувати до вхідного тексту.

Модуль об’єднання результатів класифікації та модуль аналізу результатів класифікації можна об’єднати в блок оброблення результатів, оскільки вони маніпуляції над результатами, отриманими від класифікатора та формують кінцевий результат класифікації.

Розглянемо більш детальніше архітектуру розробленого програмного забезпечення. На рис. 3.2.2 зображена загальна діаграма класів системи класифікації коротких новинних текстів.

Програмне забезпечення написана з використанням ООП і має об’єктно-орієнтовану структуру.

Синім кольором на схемі зображено інтерфейси, тобто реалізується принцип ООП щодо абстрагування. Зеленим кольором на схемі відображено конкретні реалізації модулів та імплементація згаданих вище інтерфейсів.

Для кожного великого логічного модуля створено окремі простори імен (Namespace). На рис. 3.2.3 зображена загальна діаграма просторів імен системи класифікації коротких новинних текстів.

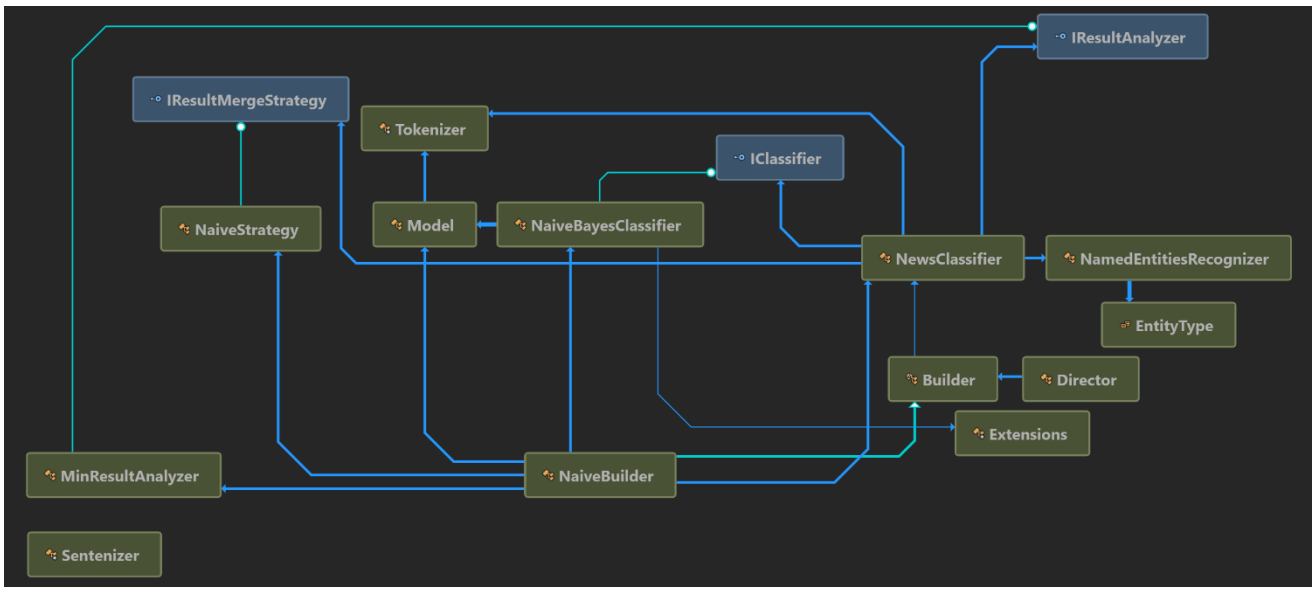

Рис. 3.2.2. Загальна діаграма класів системи класифікації коротких новинних текстів

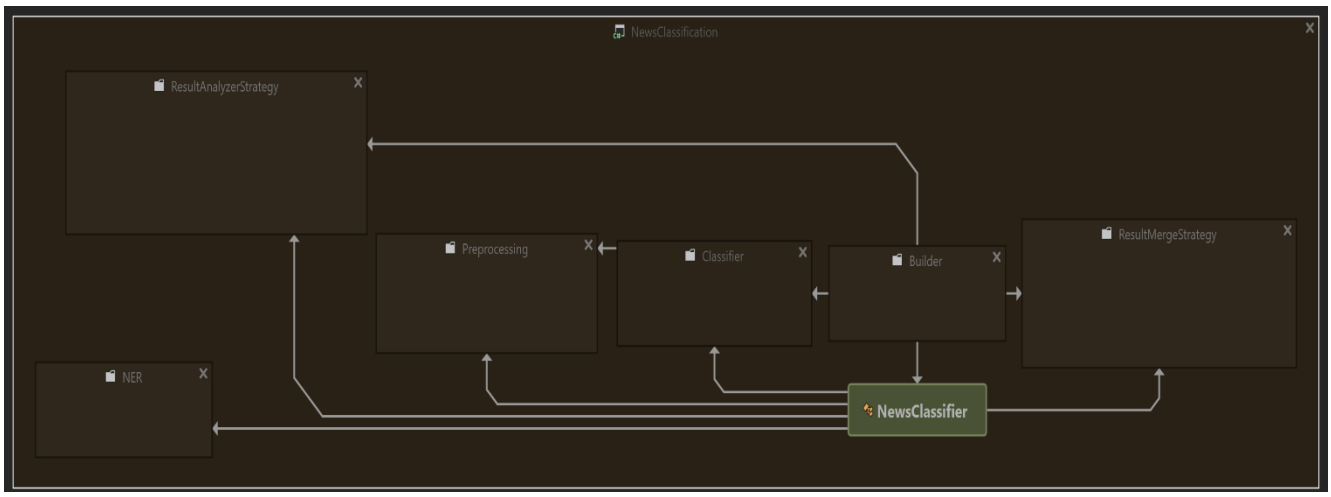

Рис. 3.2.3. Загальна діаграма просторів імен системи класифікації коротких новинних текстів

3.3 Особливості реалізації програмного забезпечення

В даному програмному застосунку було реалізовано класифікатор, класи та методи, які потрібні для:

- Токенезації.
- Лематизації.
- Виключення зайвих слів, що не несуть особливого значення для моделей класифікації.

- Прибирання зайвих «шумів» з тексту (тегів, зайвих розділових знаків, тощо).

Спираючись на проведене в першому розділі дослідження підходів та методів автоматизованої класифікації в якості класифікатора було обрано наївний Баєсів класифікатор. Проте для того, щоб забезпечити універсальність застосунку, архітектура була побудована без прив'язки до конкретної реалізації, шляхом використання абстрагування та інтерфейсів, спираючись на принципи об'єктно-орієнтованого програмування.

Лістинг 1. Інтерфес Iclassifier

```
public interface IClassifier
{
	IDictionary<string, double> Classify(string[] content);
}
```

Як вже зазначалося вище, для класифікації було використано наївний Баєсів класифікатор, який імплементації інтерфейсу IClassifier шляхом реалізації NaiveBayesClassifier, реалізація якого продемонстрована в додатку 1.

Для реалізації етапу передобробки текстів було використано готове рішення від OpenNLP, а саме методи Sentenizer та Tokenizer. Бібліотека Apache OpenNLP – це набір інструментів на основі машинного навчання для обробки тексту природної мови, які підтримують найпоширеніші завдання NLP, що потрібні для створення більш просунутих служб обробки текстів. Заме завдяки цим методом реалізовано передобробки тексту, а саме лематизації, токенізації та прибирання зайвих «шумів» з вхідного тексту новини.

Вхідний текст для класифікатора, що працює з всім текстом новини (далі Text based classifier), приходить у векторному вигляді. Для розкладення тексту в векторний вигляд було використано представлення «мішок слів» (або «bag-of-word»). Після передобробки векторне представлення тексту новини надходить на класифікацію в клас NaiveBayesClassifier.

Класифікатор, що працює з іменованими сутностями (далі NER based classifier, де NER розшифровується як Named Entities Recognition), на етапі передобробки тексту для розпізнавання іменованих сутностей використовує бібліотеку OpenNLP. В даній роботі було використано моделі, які дозволяють розпізнавати сутності наступних типів:

- Location – географічні локації та їх назви, наприклад, країни, міста, вулиці, будівлі, національні парки, тощо.
- Money – назви грошових валют, наприклад, долар США, Українська гривня, тощо.
- Organization – назви організацій, компаній та структур, наприклад, Microsoft Corporation, Apple Inc, тощо.
- Person – імена відомих людей, наприклад політиків – Дональд Трамп, спортсменів – Michael Fred, співаків – Beyonce, акторів – Leonardo DiCaprio, режисерів – Quentin Tarantino, модельєрів – Thomas Ford, тощо.

Дані типи іменованих сутностей відображені в програмі за допомогою відповідного перелічуваного типу даних в мові програмування C# – enum.

Лістинг 2. Типи іменованих сутностей, які розпізнаються в застосунку

```
public enum EntityType
{
	Location,
	Money,
	Organization,
	Person
}
```

Визначені з вхідного новинного тексту іменовані сутності також виконується за допомогою наївного баєсівського класифікатора, та відповідно класу NaiveBayesClassifier.

Для оброблення результатів, що надходять з класифікатора NaiveBayesClassifier, яким користуються Text based classifier та NER based

classifier, було використано паттерн програмування стратегія, тому що програмне забезпечення повинне забезпечувати різні варіанти алгоритму або поведінки.

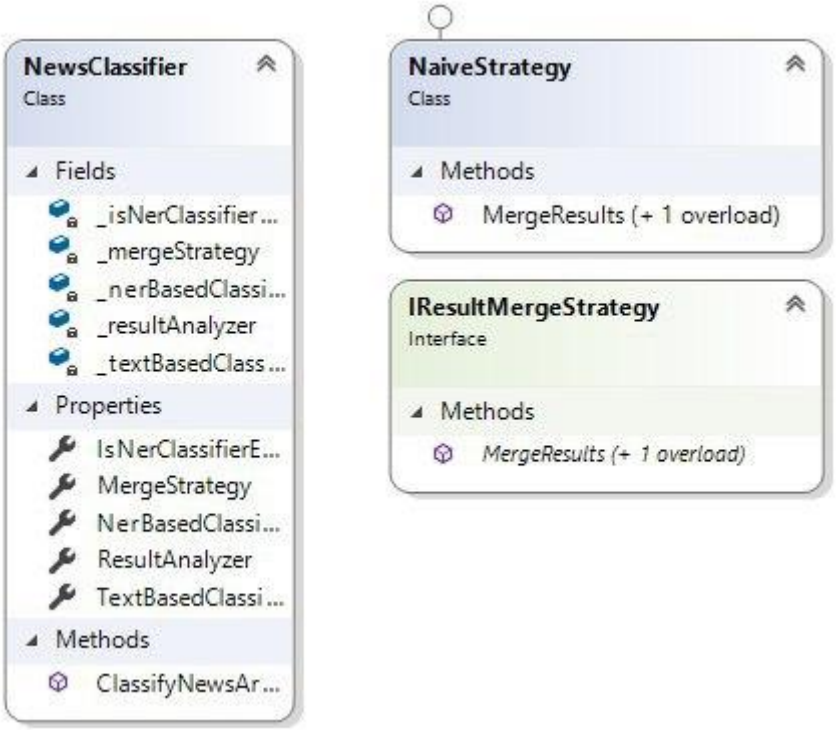

Рис. 3.3.1. Шаблон стратегія для реалізації етапу формування результатів

Стратегія (від англ. Strategy) – це поведінковий шаблон програмування також відомий як «лінія поведінки» (від англ. Policy), який визначає сімейство алгоритмів, інкапсулює кожен з них та робить їх взаємозамінними. Це поведінковий паттерн дозволяє змінювати алгоритми незалежно від коду клієнтів.

ResultMergeStrategy виконує злиття результатів класифікації декількох класифікаторів (класифікатор, що базується на повному тексті та класифікатор, що базується на іменованих сутностях). В найпростішому варіанті використовується середнє арифметичне значення для обрахунку кінцевого результату. IresultMergeStrategy має наступні сигнатури методів:

Лістинг 3. Інтерфейс IResultMergeStrategy

```
public interface IResultMergeStrategy
```

```
{
	Dictionary<string, double> MergeResults(List<IDictionary<string, double>>
results);
	Dictionary<string, double> MergeResults(params IDictionary<string,
double>[] results);
}
```

Також поведінковий паттерн стратегія був використаний для реалізації етапу аналізу результатів класифікації новинного тексту, що дозволяє прийняти кінцеве рішення, що якого класу належить текст або яку мітку (мітки) на нього необхідно поставити.

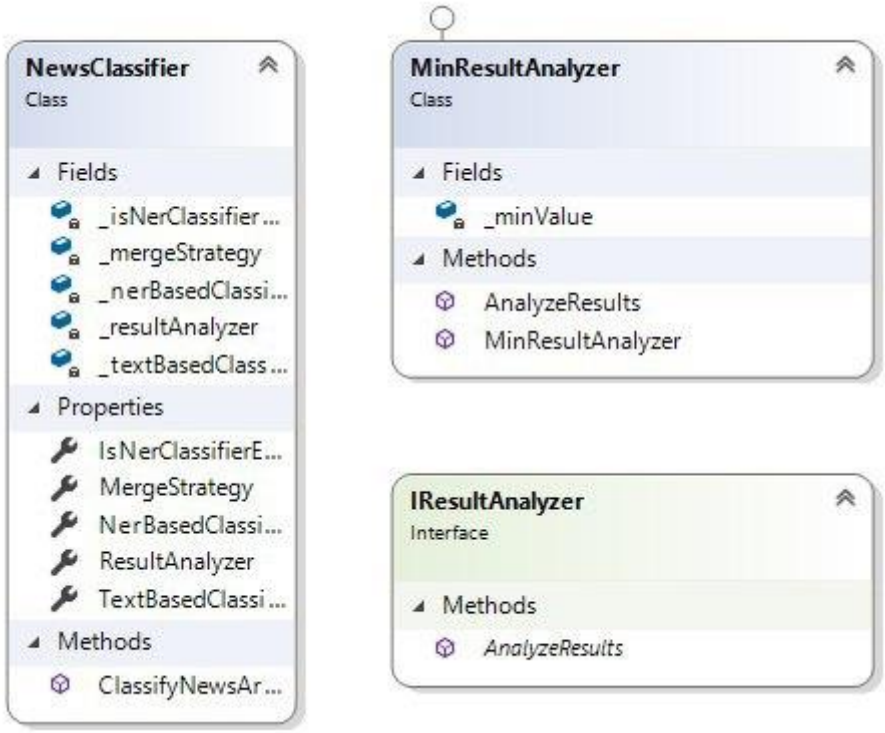

Рис. 3.3.2. Шаблон стратегія для реалізації етапу аналізу результатів

Сигнатура методу IresultAnalyzer має наступний вигляд:

Лістинг 4. Інтерфейс IresultAnalyzer

```
public interface IResultAnalyzer
{
	List<string> AnalyzeResults(IDictionary<string, double> results);
}
```

Замість створення клієнтом NewsClassifier сервісу конкретної стратегії, здійснюється передавання залежності клієнту. Ін’єкція залежності (від англ. Dependency injection) – це паттерн проектування програмного забезпечення, за допомогою якого один об’єкт (або статичний метод) постачає залежності іншого об’єкта, використовуючи «інверсію управління» (англ. Inversion of control, IoC) для розв’язання (отримання)

залежностей. Залежність – це об’єкт, який може бути використаний (сервіс). Ін’єкція – це проходження залежності до залежного об’єкта (клієнта), який його використовуватиме. Сервіс стає частиною клієнта. Існує три найбільш поширені форми впровадження залежностей:

- ін’єкція через конструктор,
- ін’єкція через властивість,
- ін’єкція через метод.

Для заміни стратегій конкретних реалізацій використовуються властивості класу NewsClassifier, тобто відбувається ін’єкція або впровадження залежності через властивість.

Лістинг 5. Властивості класу NewsClassifier, що реалізують заміну стратегій

```
public IResultMergeStrategy MergeStrategy
{
	get => _mergeStrategy;
	set => _mergeStrategy = value;
}

public IResultAnalyzer ResultAnalyzer
{
	get => _resultAnalyzer;
	set => _resultAnalyzer = value;
}
```

Використання даних стратегій наведено в лістингу 6.

Лістинг 6. Використання описаних стратегій

```
if (namedEntities.Any() && _isNerClassifierEnabled)
{
	IDictionary<string, double> nerResult =
_nerBasedClassifier.Classify(namedEntities);
	fullResult = _mergeStrategy.MergeResults(textResult, nerResult);
}
else
{
	fullResult = textResult;
}

return _resultAnalyzer.AnalyzeResults(fullResult);
```

В основу побудови програмного застосунку було покладено принципи об’єктно-орієнтованого дизайну, а самі принципу SOLID. SOLID – це п’ять базових принципів об’єктно-орієнтованого програмування та дизайну запропонована Робертом Мартіном, до якого входять принцип

єдиного обов'язку (Single responsibility principle), принцип відкритості/закритості (Open/closed principle), принцип підстановки Лісков (Liskov substitution principle), принцип розділення інтерфейсу (Interface segregation principle), принцип інверсії залежностей (Dependency inversion principle).

Так при реалізації класу NewsClassifier, було втілено принцип інверсії залежностей. Клас NewsClassifier залежить виключно від абстракцій, а не конкретних реалізацій.

Лістинг 7. Поля класу NewsClassifier, що не залежать від конкретних реалізацій

```
private IClassifier _textBasedClassifier;
private IClassifier _nerBasedClassifier;
private bool _isNerClassifierEnabled;

private IResultMergeStrategy _mergeStrategy;
private IResultAnalyzer _resultAnalyzer;
```

Для реалізації підміни абстракцій на конкретні імплементації цих абстракцій, як вже зазначалося вище, можна виконати ін'єкцію через конструктора, через властивість або через метод. У реалізації даного програмного застосунку для інкапсуляції даних залежностей використовуються властивості.

Лістинг 8. Властивості класу NewsClassifier, які використані для інкапсуляції

```
public IClassifier TextBasedClassifier
 {
     get => _textBasedClassifier;
     set => _textBasedClassifier = value;
 }
 public IClassifier NerBasedClassifier
 {
     get => _nerBasedClassifier;
     set => _nerBasedClassifier = value;
 }
 public bool IsNerClassifierEnabled
 {
     get => _isNerClassifierEnabled;
     set => _isNerClassifierEnabled = value;
 }
```

Оскільки для створення об'єкту класу NewsClassifier характерна досить складна логіка, мається на увазі що клієнт повинен створювати складені об'єкти, при цьому процес створення об'єкта можна розділити на

етапи, саме тому було вирішено використати породжуючий шаблон будівельник.

Будівельник (від англ. Builder) – це породжуючий шаблон, який відокремлює конструювання складного об’єкта від його подання (моделі), тому в результаті одного і того ж процесу конструювання можуть утворюватися різні подання.

Запропонована імплементація дозволяє приховати складність створення екземпляру класу NewsClassifier. Так Director – розпорядник: фіксований, єдиний алгоритм створення будь-яких продуктів, для яких є у наявності відповідний Builder. Builder – містить перелік всіх можливих абстрактних методів для створення та поєднання частин різноманітних об’єктів (продуктів). ConcreteBuilder – конкретний будівельник, який займається створенням якогось одного об’єкту (продукту): зв’язує разом частини продукту за допомогою реалізації тільки тих абстрактних методів інтерфейсу Builder, які потрібні для створення частин цього продукту, визначає внутрішнє подання цього складного об’єкту та надає інтерфейс для отримання останнього.

Лістинг 9. Director

```
public class Director
{
	public void Construct(Builder builder)
	{
		builder.BuildTextClassifier();
		builder.BuildNerClassifier();
		builder.BuildResultMergeStrategy();
		builder.BuildResultAnalyzerStrategy();
	}
}
```

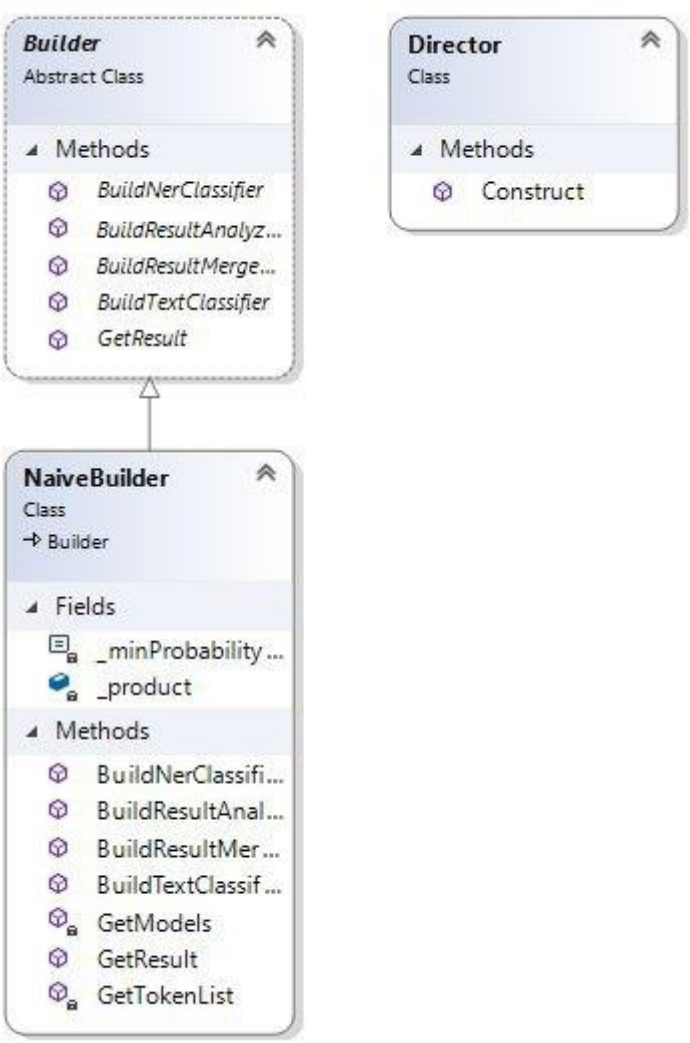

Рис. 3.3.3. Шаблон будівельник для створення класу NewsClassifier

Лістинг 10. Абстрактний Builder

```
public abstract class Builder
{
	public abstract void BuildTextClassifier();
	public abstract void BuildNerClassifier();
	public abstract void BuildResultMergeStrategy();
	public abstract void BuildResultAnalyzerStrategy();
	public abstract NewsClassifier GetResult();
}
```

Конкретний приклад використання класифікатора має спрощену структуру, ніж той, який був би без імплементації даного породжуючого шаблону програмування.

Лістинг 11. Лістинг прикладу використання коду

```
Director director = new Director();
var naiveBuilder = new NaiveBuilder();
director.Construct(naiveBuilder);

var newsClassifier = naiveBuilder.GetResult();
```

```
        const string newsText = "All the countries President Obama has visited in the
last 8 years mapped";
        List<string> classes = newsClassifier.ClassifyNewsArticle(newsText);

        Console.WriteLine($"{newsText} is classified as {string.Join(", ", classes)}");
```

3.4 Висновки

Розроблене програмне забезпечення призначене для автоматизованої класифікації коротких новинних текстів на основі запропонованого в роботі методу класифікації коротких новинних текстів з використанням іменованих сутностей.

Можна зробити наступні висновки про розроблену програмну систему в цілому:

1. Розроблену систему можна використовувати для класифікації коротких новинних текстів на будь-яку тематику.

2. Програма побудована з дотриманням принципі ООП та ООД, що забезпечує низьку зв’язаність та простоту модифікації згідно до потреб кінцевого користувача бібліотеки.

3. Архітектура розробленого програмного забезпечення дозволяє з легкістю додавати реалізацію інших методів класифікації коротких новинних текстів, а також змінювати, видаляти чи додавати певні кроки до вже реалізованого методу.

Реалізована система класифікації коротких новинних текстів з використання іменованих сутностей не залежить від інтерфейсу користувача, та може працювати з будь-яким іншим інтерфейсом чи програмним забезпеченням, як його складова частина.

4 АНАЛІЗ ЕФЕКТИВНОСТІ МЕТОДУ КЛАСИФІКАЦІЇ КОРОТКИХ НОВИННИХ ТЕКСТІВ

Для аналізу ефективності запропонованого методу класифікації коротких новинних текстів було використано розмічений набір з 2 тисяч англомовних новинних текстів, який складався з:

1. дати публікації новинного тексту;
2. заголовку;
3. тіла статті;
4. списку категорій, до яких було віднесено даний текст (від однієї і більше).

Новинний текст міг бути віднесений до однієї або декількох з даних категорій:

1. бізнес;
2. наука та технології;
3. розваги;
4. здоров’я;
5. мистецтво.

Для оцінки якості методу класифікації коротких новинних текстів використовувалися саме заголовки новин.

Як часто робиться при дослідженні якості роботи алгоритмів машинного навчання частину (50%) даних було використано для навчання алгоритму, а частину (50%) для класифікації та порівняння з еталонними даними. Також для порівняння якості роботи запропонованого алгоритму було реалізовано простий наївний Баєсів класифікатор та проведено порівняння результатів роботи обох реалізацій.

Для оцінки результатів даних алгоритмів було використано матрицю помилок та ознаки, які можуть бути розраховані на її основі.

Таблиця помилок (іноді також називається матрицею помилок) – це таблиця з двома рядками та двома стовпцями, в якій зазначається кількість помилкових спрацьовувань (FP – false positive), помилкових негативів (FN – false negative), справжніх позитивних (TP – true positive) і справжніх негативів (TN – true negative). Це дозволяє проводити більш детальний аналіз, ніж лише частка правильних класифікацій (точність). Точність не є надійною метрикою для реальної роботи класифікатора, оскільки це призведе до введення в оману результатів, якщо набір даних незбалансований (тобто коли кількість спостережень у різних класах дуже сильно відрізняється). Наприклад, якщо в наборі даних було 95 кішок і лише 5 собак, то певний класифікатор може класифікувати всі спостереження як кішки. Загальна точність становитиме 95%, але більш детально класифікатор матиме 100% розпізнавання (чутливість) для класу котів, але рівень розпізнавання 0% для класу собак. Тому на основі матриці помилок можна розрахувати деякі метрики, які показують більш повні результати аналізу.

Зазвичай матриці помилок зображують наступним чином:

$$\begin{matrix} TP & FP \\ FN & TN \end{matrix}$$

На основі даних з матриці помилок можна розрахувати наступні величини:

1. *Точність (precision)* – відношення кількість правильно розпізнаних об’єктів даного класу до всіх віднесень об’єктів до даного класу. Важливо не плутати з часткою правильно класифікованих елементів (accuracy) Якщо продовжувати приклад класифікатора котів та собак, то дана міра покаже, яка частка дійсно котів була розпізнана котом нашим класифікатором.

$$precision = \frac{TP}{TP + FP}$$

2. *Повнота(recall)* – відношення між кількістю розпізнаних об'єктів даного класу до загальної кількості об'єктів даного класу в виборці. Для класифікатора котів та собак – відношення кількості котів, які наш класифікатор правильно розпізнав до загальної кількості котів у виборці.

$$recall = \frac{TP}{TP + FN}$$

3. *F1-міра* – середнє гармонійне між *повнотою* та *точністю*. Оскільки повнота та точність в певній мірі протилежно залежні одна від одної, то для задач, в яких дані міри однаково важливі використовують *F1-міру*. Адже при суттєвому зростанні одного з елементів над іншим *F1-міра* наближується до меншого.

$$F1score = \frac{2 * precision * recall}{precision + recall} = \frac{2 * TP}{TP + TN + FP + FN}$$

4. У випадку, якщо ми можемо оцінити перевагу важливості одного зі значень використовують Fβ-міру. Де β – показник, у скільки разів повнота важливіша за точність.

$$F\beta\text{score} = \frac{(1 + \beta^2) * precision * recall}{\beta^2 * precision + recall}$$

$$= \frac{(1 + \beta^2) * TP}{(1 + \beta^2) * TP + \beta^2 * FN + FP}$$

Оскільки в нашому випадку можна вважати, що повнота і точність однаково важливі – візьмемо F1-міру

Оцінка результатів роботи проводилась для результатів по кожній категорії.

4.1 Аналіз класифікації рубрики «Бізнес»

Матриця помилок наївного Баєсового класифікатора:

$$\begin{matrix} 186 & 58 \\ 84 & 672 \end{matrix}$$

Матриця помилок запропонованого методу:

$$\begin{matrix} 204 & 40 \\ 66 & 690 \end{matrix}$$

Таблиця 1

Порівняння наївного Баєсового класифікатору та запропонованого методу класифікації коротких новинних текстів

	Наївний Баєсів класифікатор	Запропонований спосіб класифікації коротких новинних текстів
Accuracy	0,858	**0,894**
Precision	0,762	**0,836**
Recall	0,689	**0,756**
F1score	0,724	**0,794**

4.2 Аналіз класифікації рубрики «Наука та технології»

Матриця помилок наївного Баєсового класифікатора:

$$\begin{matrix} 148 & 44 \\ 70 & 738 \end{matrix}$$

Матриця помилок запропонованого методу:

$$\begin{matrix} 172 & 55 \\ 46 & 727 \end{matrix}$$

Таблиця 2

Порівняння наївного Баєсового класифікатору та запропонованого методу класифікації коротких новинних текстів

	Наївний Баєсів класифікатор	Запропонований спосіб класифікації коротких новинних текстів
Accuracy	0,886	**0,899**
Precision	**0,771**	0,758
Recall	0,679	**0,789**
F1score	0,722	**0,773**

4.3 Аналіз класифікації рубрики «Розваги»

Матриця помилок наївного Баєсового класифікатора:

$$\begin{matrix} 252 & 30 \\ 103 & 615 \end{matrix}$$

Матриця помилок запропонованого методу:

$$\begin{matrix} 264 & 24 \\ 91 & 621 \end{matrix}$$

Таблиця 3

Порівняння наївного Баєсового класифікатору та запропонованого методу класифікації коротких новинних текстів

	Наївний Баєсів класифікатор	Запропонований спосіб класифікації коротких новинних текстів
Accuracy	0,867	**0,885**
Precision	0,894	**0,917**
Recall	0,710	**0,744**
F1score	0,791	**0,821**

4.4 Аналіз класифікації рубрики «Здоров'я»

Матриця помилок наївного Баєсового класифікатора:

$$\begin{matrix} 47 & 4 \\ 45 & 904 \end{matrix}$$

Матриця помилок запропонованого методу:

$$\begin{matrix} 49 & 5 \\ 43 & 903 \end{matrix}$$

Таблиця 4

Порівняння наївного Баєсового класифікатору та запропонованого методу класифікації коротких новинних текстів

	Наївний Баєсів класифікатор	Запропонований спосіб класифікації коротких новинних текстів
Accuracy	0,951	**0,952**
Precision	**0,922**	0,907
Recall	0,511	**0,533**
F1score	0,657	**0,671**

4.5 Аналіз класифікації рубрики «Мистецтво»

Матриця помилок наївного Баєсового класифікатора:

38 7
27 928

Матриця помилок запропонованого методу:

39 7
26 928

Таблиця 5

Порівняння наївного Баєсового класифікатору та запропонованого методу класифікації коротких новинних текстів

	Наївний Баєсів класифікатор	Запропонований спосіб класифікації коротких новинних текстів
Accuracy	0,966	**0,967**
Precision	0,844	**0,848**
Recall	0,585	**0,600**
F1score	0,691	**0,703**

4.6 Загальні підсумки метрик

На більшості даних можна спостерігати невелике зростання по майже кожній з метрик на декілька відсотків (2-7%).

Також в розглянутих даних яскраво видно, що accuracy не є достатньої для оцінки якості алгоритму. Адже не дивлячись на дуже високі результати accuracy для класифікації новинних тексті для здоров’я та мистецтва з інших метрик можна побачити, що якість їх розпізнавання (особливо recall і F1score) суттєво нижча за розпізнавання для класів, які мають більшу кількість даних у виборці.

4.7 Висновки

У даному розділі проведено аналіз розробленого методу класифікації коротких новинних текстів. Метод дозволяє розв’язати задачу дослідження, даючи змогу створити автоматизовані системи з класифікації коротких новинних текстів, які можуть бути виконані при подальшій розробці ПЗ.

Проведено аналіз роботи алгоритму для кожного класу. Створено матрицю помилок та на її основі розраховано міри, які можуть бути використані для оцінки якості роботи запропонованого методу.

Розроблено програмне забезпечення призначене для аналізу запропонованого методу. В розробленому програмному забезпечені було імплементовано запропонований метод та метод, що базується на наївному Баєсовому класифікаторі.

Завдяки використанню створеного програмного забезпечення було проведено аналіз показників якості класифікації коротких новинних текстів. Запропонований метод показав кращі або рівні результати(міри відрізняються на -1% – +7%). Подальшими дослідженнями можуть бути створення вибірок з рівною кількістю об’єктів різного класу. Також

перспективним варіантом розвитку даної роботи може бути використання бітермів.

5 ПОБУДОВА БІЗНЕС-МОДЕЛІ

5.1 Опис проблеми

Розвиток Інтернету та активне впровадження цифрових технологій у повсякденне життя людини призвели до стрімкого збільшення цифрових текстових даних. З часом кількість текстових даних, в тому числі і новинних текстів, зростає і може сягати десятків і, навіть, сотень тисяч записів, що вимагає багато часу на ознайомлення з ними. Таким чином, актуальною постає проблема організації та проведення аналізу багаточисельних новинних текстів різноманітних Інтернет-видань за короткий проміжок часу без втрати повноти та точності результату аналізу. Для вирішення цієї проблеми зазвичай застосовуються методи автоматизованої класифікації текстових даних (в нашому випадку – новинних текстів мережі Інтернет).

Розроблений в даній роботі метод класифікації новинних текстів може бути широко застосований у великій кількості сфер.

Перш за все, це звичайно будуть новинні сайти, які можуть використовувати класифікацію для рубрикації новин.

У час всебічної інформатизації людського суспільства розповсюдження новин стало значно простіше, ніж навіть декілька років тому. Доступ до новин став простим за допомогою Інтернет медіа-ресурсів, зокрема їх сторінок в соціальних мереж, і як вже зазначалося вище, особливої популярності здобули короткі новинні тексти. Але навіть завдяки зменшенню обсягів новин за допомогою використання коротких повідомлень, як у Twitter, кількість новин, а відповідно і повідомлень продовжує стрімко зростати. Користувачам, що підписані на сторінки соціальних медіа, стає просто неможливо відстежувати всю сукупність новинних повідомлень.

Користувачам, що підписані на сторінки соціальних медіа, стає просто неможливо відстежувати всю сукупність новинних повідомлень.

При цьому варто зазначити, що більшість читачів не зацікавлені у всіх темах, а мають інтерес лише до певної категорій новин, наприклад спорт чи політика. Саме тому дуже важливою стає персоналізація новинної стрічки до вподобань кожного користувача шляхом визначення категорії новини та пріоритезації її появи. Таким чином, з огляду на вищезазначене, метою даної роботи стало підвищення ефективності існуючих методів класифікації коротких новинних текстів шляхом модифікації існуючих підходів до класифікації коротких новинних текстів за рахунок врахування їх особливостей.

У рубрикації є 3 основні проблеми:

1. Складність рубрикації

   Тут основними проблемами є постійне зростання кількості новин, які надходять у реальному часі. Крім того, велика частина новин може відноситись до декількох рубрик одночасно.

2. Неоднозначність рубрикації новин

   Вибір рубрики, до якої відноситься новина досить сильно залежить від людського фактору, що іноді може призводити до некоректних результатів. Також рубрики новин досить часто перетинаються, що іноді не враховується при присвоєнні рубрики новини.

3. Висока вартість рубрикації

   В більшості випадків для рубрикації новин використовується найманий персонал. Це є проблемою і на те є декілька причин: робота найманого персоналу коштує грошей, живий персонал досить погано масштабується для великого потоку новин та має обмежену швидкість обробки.

Також, досить проблематично оброблювати зміни тексту. Тому що зазвичай, новина відноситься до певної категорії тільки під час створення і зміна тексту жодним чином не моніториться.

Описані вище проблеми в свою чергу створюють незручності кінцевим користувачам новинних сайтів (читачам). Адже їм складно здійснювати пошук по стрічці новин.

Для покращення досвіду користувачів можна вирішити проблеми відсутності тегування статей. Також додати можливість фільтрувати по перетинах категорій та формувати власну стрічку на основі результатів класифікації.

Всі описані вище проблеми узагальнені у дереві проблем, що зображено на рис. 5.1.1.

5.2 Зацікавлені сторони

У вирішенні описаних вище зацікавлено певне коло осіб.

Як уже зазначалося у попередньому розділі, найбільший інтерес у вирішенні наведених вище проблем, виявляють самі новинні ресурси. Як і більшість інших медіа новинні ресурси заробляють кошти шляхом додавання рекламних матеріалів до своїх статей.

Найбільш розповсюдженим, популярним та дієвим методом реклами на сьогоднішній день є таргетована реклама. Таргетована реклама – це текстові, медійні або мультимедійні оголошення, які демонструються тільки тим користувачам мережі, які задовольняють певним набором вимог, заданому рекламодавцем [39].

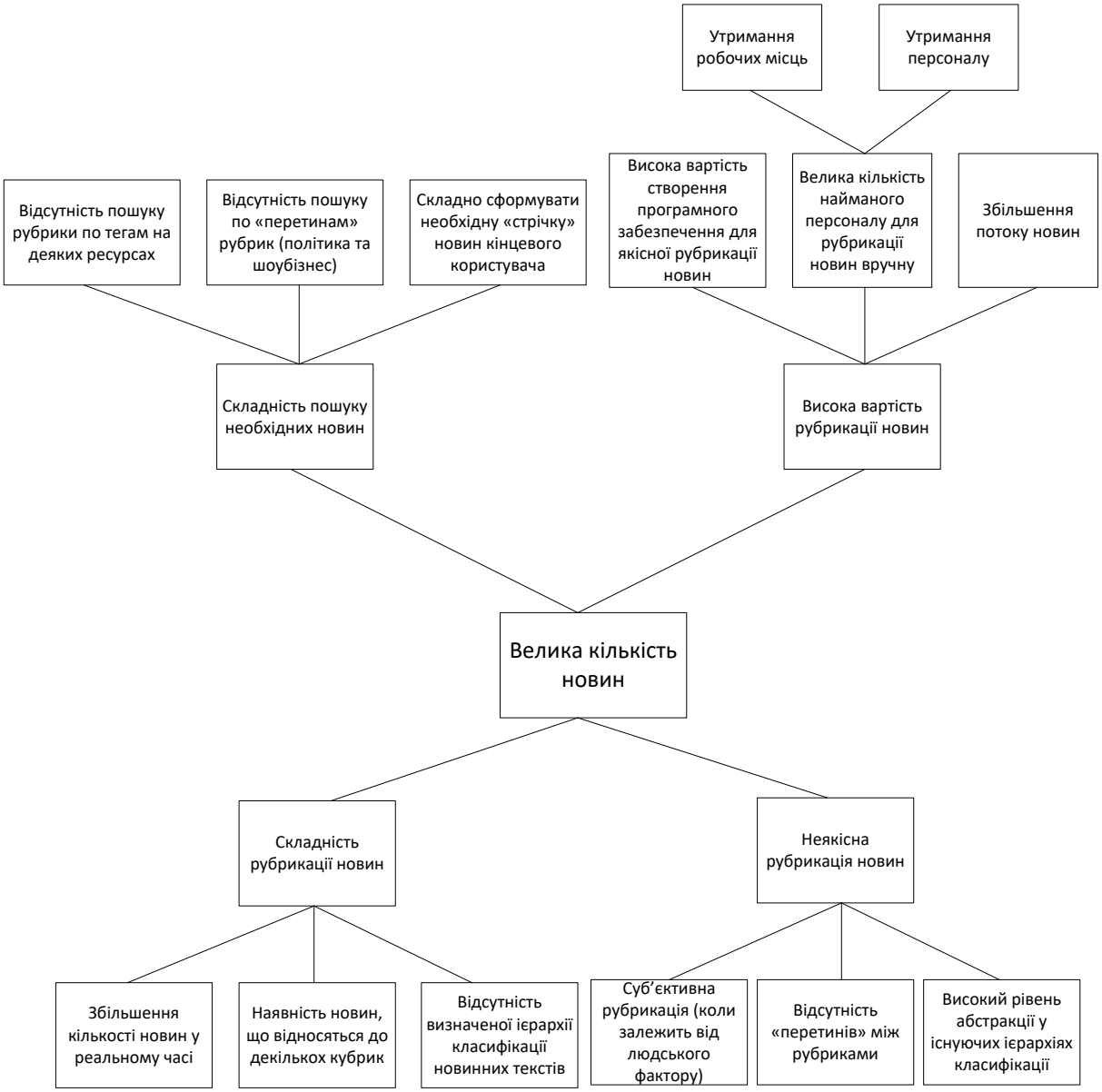

Рис. 5.1.1. Дерево проблем

Для якісного продажу майданчиків для розміщення реклами новинні сервіси повинні відповідати наступним вимогам:

- Охоплювати велику аудиторію читачів (потенційних покупців товарів та послуг, що рекламуються);
- Просувати відповідним користувачам найбільш релевантну для них рекламу, тим самим збільшуючи ймовірність покупки ними товарів чи послуг.

Забезпечити реалізацію цих вимог дозволяє розроблений метод класифікації новинних текстів. По-перше, зручна рубрикація новин надає

медіа ресурсу перевагу перед іншими аналогічними платформами, тим самим приваблюючи користувачів. По-друге, підбір правильних новин може допомогти охарактеризувати вподобання користувача та запропонувати йому найбільш цікаву рекламу. Наприклад, якщо користувач цікавиться новинами про спорт, а конкретніше бокс, то йому може рекламуватися широкий спектр пов'язаних матеріалів: спортивна форма, рукавиці, спортивне харчування, гуртки з боксу і тому подібне.

Другою зацікавленою стороною є самі читачі. Сьогодні звичайний людина отримує кожне день у п'ять разів більше інформації, ніж у 1986. Ступінь інформаційної революції та цифрового віку була розрахована доктором Мартіном Гілбертом та його командою в Університеті Південної Каліфорнії. Дослідники опрацювали 60 категорій аналогових та цифрових технологій у період з 1986 по 2007 рік, і результати відображають майже повний перехід людства до цифрової епохи. Використовуючи аналогію 85-сторінкової газети, вчені виявили, що в 1986 році людина отримувала близько 40 газет, повних інформації щодня, але це значення зростало і досягло 174 в 2007 році [40]. Зважаючи на дані тенденції сучасний читати бажає отримати лише релевантну інформацію та не витрачати свій час дарма. Зацікавленість користувачів є досить високою.

Наступною зацікавленою стороною є рекламодавці. Рекламодавці зацікавлені в ознайомленні широкої та найбільш зацікавленої аудиторії у своїх рекламних матеріалах. Саме вони є інвесторами за рахунок яких діють медіа ресурси, а відповідно і вплив та зацікавленість рекламодавців є теж високою.

Також зацікавленими сторонами у якісній рубрикації новин є піар агентства та штаби політичних партій. Зараз широко розповсюдження відслідковування реакцій населення для визначення популярності чи навпаки – недовіри певних урядових чи парламентських ініціатив, ставлення виборців до ініціатив кандидатів. Різноманітні передвиборні штаби чи просто політичні партіям могли б значно швидше моніторити

відповідні тенденції завдяки правильному аналізу новин. Крім зручності та швидкості, запропонований метод дозволив би скоротити кількість найманого персоналу та відповідних витрат на їх утримання.

Отже, зважаючи на вищесказане, можна коротко відобразити всі зацікавлені сторони (табл. 6).

5.3 Комерційне рішення. Основні характеристики

Відповідно до зазначеного вище, можна описати кінцевий продукт, що наддасть можливість розв’язати існуючі у даній сфері проблеми. Даний програмний продукт буде реалізовувати описаний у попередній розділах автоматизований метод класифікації новинних текстів з використанням тегів та ієрархічної класифікації. Даний метод дозволяє ефективно визначати клас, що якого відноситься новинний текст, та на основі даної категоризації формувати найбільш актуальний для користувача набір новин. Відчутних змін у роботі звичних новинних ресурсів не буде помітно відразу, проте, збільшення якості формування набору новин стане очевидним для користувачів після короткочасного використання методу.

Таблиця 6

Зацікавлені сторони

Зацікавлена сторона	**Інтерес зацікавленої особи**	**Вплив зацікавленої особи**	**Стратегії приваблення зацікавлених сторін**
Новинні ресурси	Економний та точний спосіб рубрикації новинних текстів	Високий	Презентація продукту. Участь у спеціалізованих конференціях та виставках. Безкоштовна тріальна версія продукту.
Читачі	Підвищення якості підбору необхідних новинних матеріалів	Середній	
Рекламодавці	Впевненість в охопленні рекламою широкої аудиторії та релевантної аудиторії	Середній	
Агрегатори новин	Економний та точний спосіб рубрикації новинних текстів	Середній	
Піар-агентства	Зменшення вартості та підвищення якості моніторингу новин	Низький	

Для власників медіа ресурсів впровадження даного методу стане відчутним відразу. По-перше, зменшиться кількість витрат на утримання найманого персоналу для рубрикації новин. По-друге, покращиться якість рекомендацій таргетованої реклами, що в свою чергу привабить більше рекламодавців та інвестицій відповідно.

Клієнтами, як споживачами продукту, що базується на даному методі класифікації новинних текстів, будуть відповідно різноманітні медіа ресурси та новинні веб-портали. Тобто першочерговою є модель бізнесу B2B – Business-to-Business (бізнес для бізнесу). А вже данні ресурси надаватимуть послуги кінцевим користувачам – читачам новин [41].

Проблем інтеграції з існуючими механізмами роботи медіа ресурсів виникати не повинно, або вони повинні швидко і легко вирішитися за допомогою невеликої кількості спеціалістів. Це можна пояснити тим, що даний метод класифікації новинних текстів дозволяє замінити людину-аналога, тобто замість модуля комунікації з людиною буде розроблено програмний модуль комунікації з даним методом.

5.4 Конкурентні переваги

Методи класифікації на основі машинного навчання є відносно новими методами, проте тематика рубрикації новинних текстів вивчається вже досить довго. Існує безліч статей та наукових праць, але дані напрацювання не набули широкого використання у сучасних новинних медіа.

Але все ж варто розглянути конкурентні переваги даного методу у контексті попередніх напрацювань та в контексті аналогів, що зараз використовуються для рубрикації.

Щодо першого пункту варто зазначити, що даний метод використовує тегування та ієрархічну класифікацію. Дані підходи не використовувалися раніше у контексті вирішення проблеми рубрикації новинних статей.

Відносно другого пункту, а саме людського аналогу, то варто зазначити, що основною конкурентною перевагою є:

- відносна об’єктивність (принаймні у аналогічних ситуаціях будуть прийматися однакові рішення),

- швидкість (аналіз природомовних текстів на обладнанні з високими технічними характеристиками здійснюється значно швидше аналогічних дій людини-аналога),
- ціна (зі зростанням об’ємів даних потрібно покращувати техніку або збільшувати її об’єми, проте це дешевше, ніж утримувати офіс та велику кількість працівників, як щоразу зростатиме).

У результаті компаніям буде достатньо «навчити» класифікатор на основі попередніх «розмічених» (віднесених до певної категорії) новин та значно скоротити свої витрати зменшенням кількості найманого персоналу. Перевагою для користувача буде якісно розмічена та швидко опублікована новина. В подальшому можна використовувати даний алгоритм для формування персональної стрічки новин користувача на основі попередніх вподобань або якоїсь фільтрації на сайті (наразі вже пропонується подібні функціональні можливості у різноманітних соціальних мережах, але це робиться не спираючись на зміст попередньо прочитаних статей, а на основі певних авторів або схожих міток – тегів).

5.5 Клієнти та сегменти ринку споживання

Для даного продукту найдоцільніше провести сегментацію за швидкістю оновлення новинної стрічки та кількістю повідомлення, які повинні оброблятися за одиницю часу. Та розділити продукти на категорії «швидкого» та «повільного». Таким чином користувачі, яким не потрібно оброблювати велику кількість інформації режимі реального часу можна буде зекономити на своєму рішенні.

Також клієнтів потрібно поділити за мовною ознакою, оскільки реалізація і точність кінцевого продукту тісно пов’язана з мовою для якої було розроблено рішення. В продукті планується підтримка класифікації текстів на 3 мовах: англійська, російська та українська. На початкових стадіях продажу продукту або дослідження ринку можна конкретніше

зорієнтуватися в пріоритетах кожного з продуктів. Крім того, доцільно розробити певні регіональні ціни, адже бюджети новинних сайтів часто досить сильно залежать від націленості новин на регіональну або міжнародну аудиторію.

5.6 Унікальна ціннісна пропозиція

Ціннісна пропозиція описує ті переваги, які вирішують частково або у повній мірі вищеописані проблеми споживачів. Тобто ціннісна пропозиція – це пояснення того, як продукт вирішує проблему. У попередніх пунктах вже було детально описано проблеми та шляхи їх вирішення для різних категорій зацікавлених користувачів шляхом використання методу класифікації новинних статей. Тому у цьому розділі пропонується зосередитися на іншому тлумаченні унікальної ціннісної пропозиції, а саме: унікальною буде така пропозиція, яка чітко відділяє продукт від конкурентів. Як уже зазначалося у пункті 5.4, даний метод використовує тегування для покращення рубрикації новинних текстів. Так, наприклад, тег «Барак Обама» може бути віднесеним до рубрик «Політика» та «Знаменитості». Даний підхід дозволяє знаходити споріднені новини не лише в одній категорій, а і в інших, тим самим знаходити новини, що знаходяться на межі двох категорій.

5.7 Доходи і витрати

До загальних витрат відносяться витрати на оренду приміщення, комунальних послуг та витрат на засоби зв’язку. Ще однією з суттєвих статей витрат є витрати пов’язані з рекламою та маркетингом та оплату роботи найманого персоналу. Детальніше з планом витрат можна ознайомитися з таблиць (табл. 7, 8).

Таблиця 7

Заробітна плата персоналу

Список посад	**Кількість робітників**	**Щомісячна зп, $**	**Період роботи**	**ЗП за посадами (в місяць) , $**	**ЗП за посадами з податками(в місяць), $**
Генеральний директор	1	3000	Весь проект	3000	4551
Директор з розвитку	1	4500	Весь проект	4500	6826
Технічний директор	1	3500	Весь проект	3500	5310
Менеджер проекту	1	2500	Весь проект	2500	3793
Бухгалтер	1	300	Весь проект	300	455
Програміст (джуніор)	2	500	Весь проект	1000	1050
Програміст (сініор)	2	2500	Весь проект	5000	5250
Системний адміністратор	1	600	Весь проект	600	910
Прибиральник	1	50	Весь проект	50	76
Результат:	**11**			**17450**	**28221**

Таблиця 8

Загальні витрати

Список витрат	**Сума в місяць , $**	**Періодичність**
Оренда	1200	Місяць
Реклама та маркетинг	2000	Місяць
Комунальні послуги	100	Місяць
Оплата послуг інтернет-провайдерів	120	Місяць
Оплата послуг операторів мобільного зв’язку	50	Місяць
Оплата інших послуг (хостинг, домене ім’я)	130	Місяць
Результат:	**3600**	

Таблиця 9

Витрати на реалізацію проекту

Найменування витрат	**1-й місяць, т. $**	**2-й місяць, т. $**	**3-й місяць, т. $**	**4-й місяць, т. $**	**5-й місяць, т. $**	**6-й місяць, т. $**	**7-й місяць, т. $**	**8-й місяць, т. $**	**Загальні результати, т. $**
Загальні витрати	1.6	1.6	3.6	3.6	3.6	3.6	3.6	3.6	24.8
ЗП		28.2	28.2	28.2	28.2	28.2	28.2	28.2	197.4
Витрати	**1.6**	**29.8**	**31.8**	**31.8**	**31.8**	**31.8**	**31.8**	**31.8**	**222.2**
Заплановані прибутки					**94**	**94**	**94**	**94**	**376**
Результат(без оподаткування):	**-1.6**	**-29.8**	**-31.8**	**-31.8**	**62.2**	**62.2**	**62.2**	**62.2**	**153.8**

Прибутки очікуються на п’ятому місяці від продажу продукту. Детальніше ознайомитися зі зведеним планом прибутків та витрат можна з таблиці (табл. 9).

5.8 Бізнес модель

Узагальнимо, все написане вище у лаконічну бізнес-модель у вигляді lean canvas (табл. 10).

Таблиця 10

Канва бізнес-моделі

<table>
<tr><td rowspan="2">Проблема

Складність рубрикації

Неоднозначність рубрикації новин

Висока вартість рубрикації</td><td>Рішення

програмне забезпечення, що виконує автоматизовану класифікацію новинних текстів</td><td rowspan="2">Унікальна ціннісна пропозиція

Надання тегування

Надання ієрархічної класифікації</td><td>Прихована перевага

Можливість створювати «кастомні» класифікатори для клієнтів під замовлення</td><td rowspan="2">Споживачі

Новинні сайти та агрегатори новин</td></tr>
<tr><td>Ключові метрики

Кількість проданих одиниць продукту кожного виду</td><td>Канали

Відділи закупок новинних сайтів та агрегаторів</td></tr>
<tr><td colspan="3">Структура витрат

Утримання персоналу
Утримання офісу
Податкові витрати</td><td colspan="2">Потоки доходів

Доходи від продажу продукту</td></tr>
</table>

Отже, зважаючи на написане вище, можна зробити висновок, що даний продукт можна реалізувати та на основі цього побудувати в подальшому бізнес. Наведені розрахунки не є точними, адже не врахову.ть всіх ризиків та специфіки оподаткування, проте побудована бізнес-модель вказує на життєздатність проекту.

5.9 Висновки

У даному розділі було розглянуто основні проблеми, що існують у сфері класифікації новинних текстів, та підсумовано їх у дереві проблем. Також були розглянуті зацікавлені сторони у вирішенні даних проблем та

визначення їх впливу на подальше вирішення даних проблем, проведено дослідження потенційних клієнтів та сегменту ринку споживання.

У якості комерційного рішення було запропоновано програмний продукт, який базується на запропонованому у даній дисертації методі класифікації новинних текстів, що вирішує наявні проблеми та враховує інтерес зацікавлених осіб.

Було визначено конкурентні переваги та унікальну ціннісну пропозицію запропонованого продукту. На основі наведених вище досліджень було спрогнозовано потенційні доходи і витрати. Як результат була сформована бізнес-модель, що доводить потенціал даного продукту.

ВИСНОВКИ

У даній роботі було проаналізовано існуючі підходи та методи автоматизованої класифікації текстових даних загалом та коротких новинних текстів зокрема, визначено їхні основні переваги та недоліки.

Проведено вивчення специфіки коротких новинних загалом та їх особливості в публікаціях в соціальних мережах.

Проведено аналіз процесу класифікації коротких новинних текстів, а саме підвищення ваги іменованих сутностей в тексті.

Розроблено метод класифікації коротких новинних текстів, що пропонує використання двох класифікаторів для класифікації коротких новинних текстів.

Реалізовано програмне забезпечення для класифікації коротких новинних тексів, яке реалізує запропонований метод та існуючий метод для порівняння.

Проведено тестування розробленої системи на відібраній та розміченій вибірці з 2 тисяч новинних текстів.

Розроблена системи автоматизованого визначення образливого вмісту, що базується на запропонованому методі, показала кращі або рівні результати (міри відрізняються на -1% – +7%). Подальшими дослідженнями можуть бути.

Перспективний напрямком подальших досліджень автор вважає створення вибірок з рівною кількістю об’єктів різного класу, розробку стратегій об’єднання результатів класифікації та прийняття рішення щодо мітки. Також цікавим варіантом подальших досліджень є використання бітермів.

СПИСОК ВИКОРИСТАНИХ ЛІТЕРАТУРНИХ ДЖЕРЕЛ

1. Welcome to the information age – 174 newspapers a day: The telegraph. [Електронний ресурс] — Режим доступу: http://www.telegraph.co.uk/news/science/science-news/8316534/Welcome-to-the-information-age-174-newspapers-a-day.html (дата звернення 09.10.2017). — Назва з екрана.

2. *Ina Blau, Avner Caspi* Studying Invisibly: Media Naturalness and Learning / *Blau Ina, Caspi Ayner* // Evolutionary Psychology and Information Systems Research, Volume 24 of the series Integrated Series in Information Systems. — 2010. — Pp. 193-216

3. Statistical Test [Електронний ресурс] — Режим доступу: http://mathworld.wolfram.com/StatisticalTest.html (дата звернення 30.03.2018). — Назва з екрана.

4. *Har-Peled, S., Roth, D., Zimak, D.* (2003) «Constraint Classification for Multiclass Classification and Ranking.» In: Becker, B., Thrun, S., Obermayer, K. (Eds) Advances in Neural Information Processing Systems 15: Proceedings of the 2002 Conference, MIT Press. ISBN 0-262-02550-7

5. Library of Congress (2008). The subject headings manual. Washington, DC.: Library of Congress, Policy and Standards Division. (Sheet H 180: «Assign headings only for topics that comprise at least 20% of the work.»)

6. Soergel Dagobert Organizing information: Principles of data base and retrieval systems. — Orlando, FL: Academic Press, 1985. — p. 230.

7. M. Lobur Defining an approach for deep sentiment analysis of reviews in ukrainian / Lobur M., Romaniuk A., Romanyshyn M. // 2012 Комп'ютерні системи проектування теорія і практика №747, Вісник Національного університету «Львівська політехніка». — 2012. — Режим доступу: http://ena.lp.edu.ua:8080/xmlui/bitstream/handle/ntb/23309/24-124-130.pdf.

8. *Анна Пазельская* Метод определения эмоций в текстах на русском языке / *Пазельская Анна, Соловьев Алексей* // The international conference on computational linguistics and intellectual technologies «Dialogue 2011» : конференция. — Москва, 2011. — С. 512.

9. *М. В. Клековкина* Метод автоматической классификации текстов по тональности, основанный на словаре эмоциональной лексики (рус.) / *Клековкина М. В.*, Котельников Е.В. // RCDL-2012, Переславль-Залесский, Россия : конференция. — 2012. — С. 81. — Режим доступу: http://ceur-ws.org/Vol-934/paper15.pdf.

10. Internet World Stats [Електронний ресурс] — Режим доступу: https://www.internetworldstats.com/stats.htm (дата звернення 30.03.2018). — Назва з екрана.

11. Derek Thompson «Upworthy: I Thought This Website Was Crazy, but What Happened Next Changed Everything». The Atlantic [Електронний ресурс] — Режим доступу: https://www.theatlantic.com/business/archive/2013/11/upworthy-i-thought-this-website-was-crazy-but-what-happened-next-changed-everything/281472/ (дата звернення 14.11.2013). — Назва з екрана.

12. Katy Waldman «Mind the 'curiosity gap': How can Upworthy be 'noble' and right when its clickbait headlines feel so wrong?». National Post. [Електронний ресурс] — Режим доступу: http://nationalpost.com/news/mind-the-curiosity-gap-how-can-upworthy-be-noble-and-right-when-its-clickbait-headlines-feel-so-wrong (дата звернення 23.05.2014). — Назва з екрана.

13. Emily Shire «Saving Us From Ourselves: The Anti-Clickbait Movement». The Daily Beast. [Електронний ресурс] — Режим доступу: https://www.thedailybeast.com/saving-us-from-ourselves-the-anti-clickbait-movement (дата звернення 14.07.2014). — Назва з екрана.

14. Ingram, Mathew «The internet didn’t invent viral content or clickbait journalism — there’s just more of it now, and it happens faster». GigaOM. [Електронний ресурс] — Режим доступу: https://gigaom.com/2014/04/01/the-

internet-didnt-invent-viral-content-or-clickbait-journalism-theres-just-more-of-it-now-and-it-happens-faster/ (дата звернення 14.04.2014). — Назва з екрана.

15. «Hypertext» (definition). Marriam-webster Free Online Dictionary. [Електронний ресурс] — Режим доступа: https://www.merriam-webster.com/dictionary/dictionary (дата звернення 26.02.2015). — Назва з екрана.

16. McTear, Michael, Callejas, Zoraida, Griol Barres, David The Conversational Interface — Springer International Publishing., 2016. — p. 167.

17. А. И. Кострикин, Ю. И. Манин. Линейная алгебра и геометрия. Учебное пособие для вузов. — 2-е изд., перераб. — М.: Наука, Главная редакция физико-математической литературы, 1986. — 304 с.: ил.

18. *І. А. Жуков* Динамічна просторово-логічна кластеризація нейронної мережі. / Жуков І. А., Кременецький Г. М.// МІЖНАРОДНИЙ НАУКОВО-ТЕХНІЧНИЙ ЖУРНАЛ «ІНФОРМАЦІЙНІ ТЕХНОЛОГІЇ ТА КОМП'ЮТЕРНА ІНЖЕНЕРІЯ» — 2009. — Режим доступу: http://www.nbuv.gov.ua/old_jrn/natural/Itki/2009_1/09ziatnn.pdf.

19. *Leo Breiman* Random Forests / Breiman Leo // Machine Learning — Volume 45, Issue 1, October 2001. — 2001. — Pp. 5-32.

20. Кирия И. В. Что такое мультимедиа? // Журналистика и конвергенция: почему икак традиционные СМИ превращаются в мультимедийные. / под ред. А. Г. Качкаевой. М.

21. Шилина М. Г. Интернет-коммуникация и теоретические аспекты исследований масс-медиа. [Електронний ресурс] — Режим доступа: http://www.mediascope.ru/node/972 (дата звернення 29.11.2017). — Назва з екрана.

22. Доктор К. Ньюсономика: Двенадцать трендов, которые изменят новости. — М., 2013. — С. 211

23. Bits & pieces from the Online News Association's annual event in Los Angeles [Електронний ресурс] — Режим доступа: https://medium.com/3-

to-read/the-very-unofficial-blog-of-ona15-2cd15565278c#.neziw74pf (дата звернення 09.03.2018). — Назва з екрана.

24. World Press Trends 2015 [презентація]. [Електронний ресурс] — Режим доступа: http://www.wan-ifra.org/sites/default/files/field_message_file/250515%20WPT%202015%20Final.pdf (дата звернення 01.05.2018). — Назва з екрана.

25. Rich C. Writing and Reporting News. A Coaching Method. Wadsworth, Cengage Learning, 2007, 2010. C. 18.

26. Eye-Track Research [Електронний ресурс] — Режим доступа: https://pegiestarkadam.com/research/ (дата звернення 12.04.2018). — Назва з екрана.

27. A study of print and online newsreading[презентація]. [Електронний ресурс] — Режим доступа: http://manuscritdepot.com/internet-litteraire/document-pdf.01/infopresse/avenir-imprime/Eyetrack07ASNE.pdf (дата звернення 29.03.2018). — Назва з екрана.

28. Clark R. P. Ways to make room for good writing on social networks. [Електронний ресурс] — Режим доступа: http://www.poynter.org/2010/five-ways-to-make-room-for-good-writing-onsocial-networks/106941/ (дата звернення 01.04.2018). — Назва з екрана.

29. Support Twitter. [Електронний ресурс] — Режим доступа: https://support.twitter.com/articles/20169441 (дата звернення 11.01.2018). — Назва з екрана.

30. TIOBE Index for May 2018 [Електронний ресурс] – Режим доступу: https://www.tiobe.com/tiobe-index/ – (09.10.2017).

31. Microsoft [Електронний ресурс] – Режим доступу: https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/– (02.02.2018).

32. Microsoft [Електронний ресурс] – Режим доступу: https://blogs.msdn.microsoft.com/dotnet/2016/08/24/whats-new-in-csharp-7-0/– (02.02.2018).

33. Roslyn [Електронний ресурс] – Режим доступу: https://github.com/dotnet/roslyn – (29.04.2018).

34. Microsoft [Електронний ресурс] – Режим доступу: https://docs.microsoft.com/en-us/dotnet/framework/get-started/net-core-and-open-source/– (12.05.2018).

35. The Jython Project [Електронний ресурс]. — Режим доступу: http://www.jython.org/

36. IronPython [Електронний ресурс]. — Режим доступу: http://ironpython.codeplex.com/

37. tinypy.org [Електронний ресурс]. — Режим доступу: http://www.tinypy.org/

38. The Computer Language Benchmarks Game [Електронний ресурс]. — Режим доступу: http://benchmarksgame.alioth.debian.org/u64q/compare.php?lang=python3&lang2=gpp

39. «Записки маркетолога». – Режим доступу: http://www.marketch.ru/marketing_dictionary/marketing_terms_t/targeting/

40. Welcome to the information age – 174 newspapers a day: [The telegraph]. – Режим доступу: http://www.telegraph.co.uk/news/science/science-news/8316534/Welcome-to-the-information-age-174-newspapers-a-day.html

41. «Записки маркетолога». – Режим доступу: http://www.marketch.ru/marketing_dictionary/marketing_terms_b/b2b/

ДОДАТКИ

Додаток 1

Лістинги

```
namespace NewsClassification.Builder
{
    public abstract class Builder
    {
        public abstract void BuildTextClassifier();
        public abstract void BuildNerClassifier();
        public abstract void BuildResultMergeStrategy();
        public abstract void BuildResultAnalyzerStrategy();
        public abstract NewsClassifier GetResult();
    }
}
```

```
namespace NewsClassification.Builder
{
    public class Director
    {
        public void Construct(Builder builder)
        {
            builder.BuildTextClassifier();
            builder.BuildNerClassifier();
            builder.BuildResultMergeStrategy();
            builder.BuildResultAnalyzerStrategy();
        }
    }
}
```

```
namespace NewsClassification.Builder
{
    using System.Collections.Generic;
    using System.IO;
    using System.Linq;
    using Classifier.NaiveBayesClassifier;
    using ResultAnalyzerStrategy;
    using ResultMergeStrategy;

    public class NaiveBuilder: Builder
    {
        private const double _minProbabilityValue = 0.25;

        private NewsClassifier _product = new NewsClassifier();

        public override void BuildTextClassifier()
        {
            Dictionary<string, string> models = GetModels("/textBased");
            IEnumerable<Model> loadedModels = models.Select(x => new
Model(x.Key, x.Value));

            HashSet<string> loadedIgnoredWords =
GetTokenList("ignoredWords");

            _product.TextBasedClassifier = new
NaiveBayesClassifier(loadedModels, loadedIgnoredWords);
        }

        public override void BuildNerClassifier()
        {
            Dictionary<string, string> models = GetModels("/nerBased");
            IEnumerable<Model> loadedModels = models.Select(x => new
Model(x.Key, x.Value));
```

```
            var loadedIgnoredWords = new HashSet<string>();

            _product.TextBasedClassifier = new
NaiveBayesClassifier(loadedModels, loadedIgnoredWords);
            _product.IsNerClassifierEnabled = true;
        }

        public override void BuildResultMergeStrategy()
        {
            _product.MergeStrategy = new NaiveStrategy();
        }

        public override void BuildResultAnalyzerStrategy()
        {
            _product.ResultAnalyzer = new
MinResultAnalyzer(_minProbabilityValue);
        }

        public override NewsClassifier GetResult()
        {
            return _product;
        }

        private static Dictionary<string, string> GetModels(string path)
        {
            string[] files = Directory.GetFiles(path);

            return files.ToDictionary(x => x, x => Path.Combine(path, x));
        }

        private static HashSet<string> GetTokenList(string filename)
        {
            var result = new HashSet<string>();

            if (!File.Exists(filename))
                throw new FileNotFoundException("Unable to find Excluded
Files List at " + filename);

            using (StreamReader file = new StreamReader(filename))
            {
                string line;
                while ((line = file.ReadLine()) != null)
                {
                    if (string.IsNullOrWhiteSpace(line.Trim())) continue;

                    if (line.StartsWith("--")) continue;

                    if (!result.Contains(line.Trim()))
                        result.Add(line);
                }
            }

            return result;
        }
    }
}



namespace NewsClassification.Classifier.NaiveBayesClassifier
{
    using System;
```

```
    using System.Collections.Generic;
    using System.Globalization;
    using System.IO;
    using System.Linq;
    using Preprocessing;

    public class Model
    {
        private const char _separator = ',';

        private readonly Dictionary<string, double> _words;
        private readonly string _modelFileName;

        private long _totalWords;
        private readonly string _className;
        private readonly bool _saveModel;

        public string Name => _className;

        public long TotalWords => _totalWords;

        public Model(string className, string modelFilePath, bool loadModel
= true, bool saveModel = false)
        {
            if (string.IsNullOrWhiteSpace(className))
                throw new Exception("Class name was not defined");

            _saveModel = saveModel;
            _className = className;
            _words = new Dictionary<string,
double>(StringComparer.OrdinalIgnoreCase);
            _modelFileName = modelFilePath;
            if (loadModel)
                LoadEvidenceFromCache(modelFilePath);
        }

        ~Model()
        {
            if (_saveModel)
                SaveModel();
        }

        public void LoadEvidenceFromCache(string filePath)
        {
            if (!File.Exists(filePath))
                throw new ArgumentNullException(nameof(filePath), "File
should exist");


            using (StreamReader file = new StreamReader(filePath))
            {
                string line;
                while ((line = file.ReadLine()) != null)
                {
                    string[] keyValue = line.Split(_separator);
                    if (!_words.ContainsKey(keyValue[0]))
                    {
                        long value = long.Parse(keyValue[1]);
                        _words.Add(keyValue[0], value);
                        _totalWords += value;
                    }
                    else
                    {
```

```
                        throw new Exception(
                            $"Duplicate entries of {keyValue[0]} found
while loading model from {filePath}");
                    }
                }
            }
        }

        public bool AddData(string trainingData, HashSet<string>
wordsToIgnore)
        {
            if (string.IsNullOrWhiteSpace(trainingData)) return false;

            List<string> tokens =
Tokenizer.TokenizeNow(trainingData).ToList();

            foreach (
                string token in
                tokens.Where(token => !string.IsNullOrWhiteSpace(token) &&
!wordsToIgnore.Contains(token)))
            {
                if (!_words.ContainsKey(token))
                    _words[token] = 0;

                _words[token]++;
                _totalWords++;
            }

            return true;
        }

        public bool AddData(IEnumerable<string> trainingData,
HashSet<string> wordsToIgnore)
        {
            foreach (string data in trainingData)
            {
                AddData(data, wordsToIgnore);
            }

            return true;
        }

        public IDictionary<string, double> GetModel()
        {
            return _words;
        }

        public bool SaveModel(bool backupExisting = true)
        {
            if (_words == null || _words.Count <= 0)
                return false;

            if (backupExisting)
                BackupFile(_modelFileName);

            using (StreamWriter file = new StreamWriter(_modelFileName))
            {
                foreach (
                    string line in
                    _words.Select(
                        evidence =>
                            evidence.Key.Trim() + _separator +
evidence.Value.ToString(CultureInfo.InvariantCulture).Trim())
```

```
                )
                {
                    file.WriteLine(line);
                }
            }

            return true;
        }

        public static void BackupFile(string fullFileName)
        {
            if (!File.Exists(fullFileName))
            {
                throw new ArgumentNullException(nameof(fullFileName), "File
should exist");
            }

            string ext = Path.GetExtension(fullFileName);
            string newFileName = fullFileName +
DateTime.Now.ToString(".yyyy.MM.dd.HH.mm.ss.fff") + ext;
            File.Copy(fullFileName, newFileName);
        }
    }

}



namespace NewsClassification.Classifier.NaiveBayesClassifier
{
    using System;
    using System.Collections.Generic;
    using System.Linq;

    public class NaiveBayesClassifier : IClassifier
    {
        private const int _chunkSize = 50;

        private HashSet<string> _skippedWords;
        public IReadOnlyList<Model> Models;

        public NaiveBayesClassifier(IEnumerable<Model> models,
HashSet<string> skippedWords)
        {
            _skippedWords = skippedWords;
            Models = (IReadOnlyList<Model>) models;
        }

        public IDictionary<string, double> Classify(string[] content)
        {
            if (Models.Count < 2)
            {
                throw new Exception("List of models can't have less than 2
elements");
            }

            double chunkSize = Math.Ceiling(content.Length /
(double)_chunkSize);

            var scores = new List<Dictionary<string, double>>();
            for (int i = 0; i < chunkSize; i++)
            {
```

```
                Dictionary<string, double> score =
Models.ToDictionary<Model, string, double>(model => model.Name, evidence =>
0);

                scores.Add(score);
            }
            int index = 0;
            foreach (IEnumerable<string> wordsChunk in
content.Chunk(_chunkSize))
            {
                foreach (
                    string word in
                        wordsChunk.Where(word =>
!string.IsNullOrWhiteSpace(word) && !_skippedWords.Contains(word)))
                {
                    foreach (var modelItem in Models)
                    {
                        IDictionary<string, double> model =
modelItem.GetModel();
                        double wordCount;
                        wordCount = model.TryGetValue(word, out wordCount)
                            ? wordCount
                            : 0.01;

                        double score = Math.Log(wordCount /
modelItem.TotalWords);
                        scores[index][modelItem.Name] += score;

                    }
                }

                long totalWordsAllCategories = Models.Sum(x =>
x.TotalWords);

                foreach (var model in Models)
                {
                    scores[index][model.Name] += Math.Log((double)
model.TotalWords / totalWordsAllCategories);
                }

                List<double> scoresPerModel = Models.Select(model =>
Math.Exp(scores[index][model.Name])).ToList();

                double totalScore = scoresPerModel.Sum();

                for (int i = 0; i < Models.Count; i++)
                {
                    scores[index][Models[i].Name] = scoresPerModel[i] /
totalScore;
                }
                index++;
            }

            Dictionary<string, double> results = Models.ToDictionary<Model,
string, double>(model => model.Name, evidence => 0);

            foreach (Dictionary<string, double> score in scores)
            {
                foreach (var model in Models)
                {
                    results[model.Name] += score[model.Name];
                }
            }
```

```
            foreach (var model in Models)
            {
                results[model.Name] = results[model.Name] / scores.Count;
            }

            return results;
        }

    }

}



namespace NewsClassification.Classifier
{
    using System;
    using System.Collections.Generic;
    using System.Diagnostics;

    public static class Extensions
    {
        public static IEnumerable<IEnumerable<T>> Chunk<T>(this
IEnumerable<T> source,
                                                           int chunkSize)
        {
            if (source == null)
                throw new ArgumentNullException(nameof(source));
            if (chunkSize <= 0)
                throw new ArgumentOutOfRangeException(nameof(chunkSize),
                                                      "The chunkSize
parameter must be a positive value");

            return source.ChunkInternal(chunkSize);
        }

        private static IEnumerable<IEnumerable<T>> ChunkInternal<T>(
            this IEnumerable<T> source, int chunkSize)
        {
            Debug.Assert(source != null);
            Debug.Assert(chunkSize > 0);

            using (IEnumerator<T> enumerator = source.GetEnumerator())
                do
                {
                    if (!enumerator.MoveNext())
                        yield break;

                    yield return ChunkSequence(enumerator, chunkSize);
                } while (true);
        }

        private static IEnumerable<T> ChunkSequence<T>(IEnumerator<T>
enumerator,
                                                       int chunkSize)
        {
            Debug.Assert(enumerator != null);
            Debug.Assert(chunkSize > 0);

            int count = 0;

            do
```

```
            {
                yield return enumerator.Current;
            } while (++count < chunkSize && enumerator.MoveNext());
        }
    }
}



namespace NewsClassification.Classifier
{
    using System.Collections.Generic;

    public interface IClassifier
    {
        IDictionary<string, double> Classify(string[] content);
    }
}



namespace NewsClassification.NER
{
    public enum EntityType
    {
        Location,
        Money,
        Organization,
        Person
    }
}



namespace NewsClassification.NER
{
    using System;
    using System.Collections.Generic;
    using System.Linq;
    using java.io;
    using opennlp.tools.namefind;
    using opennlp.tools.util;

    public class NamedEntitiesRecognizer
    {
        private const string _locationModelFile = "en-ner-location.bin";
        private const string _moneyModelFile = "en-ner-money.bin";
        private const string _organizationModelFile = "en-ner-
organization.bin";
        private const string _personModelFile = "en-ner-person.bin";

        private static readonly NameFinderME LocationFinder;
        private static readonly NameFinderME MoneyFinder;
        private static readonly NameFinderME OrganizationFinder;
        private static readonly NameFinderME PersonFinder;

        static NamedEntitiesRecognizer()
        {
            var locationModel = new TokenNameFinderModel(new
FileInputStream(_locationModelFile));
            var moneyModel = new TokenNameFinderModel(new
FileInputStream(_moneyModelFile));
```

```
            var organizationModel = new TokenNameFinderModel(new
FileInputStream(_organizationModelFile));
            var personModel = new TokenNameFinderModel(new
FileInputStream(_personModelFile));

            LocationFinder = new NameFinderME(locationModel);
            MoneyFinder = new NameFinderME(moneyModel);
            OrganizationFinder = new NameFinderME(organizationModel);
            PersonFinder = new NameFinderME(personModel);
        }

        public static IEnumerable<string> GetNamedEntities(string[] tokens,
EntityType entityType)
        {
            NameFinderME nameFinder;
            switch (entityType)
            {
                case EntityType.Location:
                    nameFinder = LocationFinder;
                    break;
                case EntityType.Money:
                    nameFinder = MoneyFinder;
                    break;
                case EntityType.Organization:
                    nameFinder = OrganizationFinder;
                    break;
                case EntityType.Person:
                    nameFinder = PersonFinder;
                    break;
                default:
                    throw new
ArgumentOutOfRangeException(nameof(entityType), entityType, null);
            }
            Span[] res = nameFinder.find(tokens);
            nameFinder.clearAdaptiveData();
            return Span.spansToStrings(res, tokens).AsEnumerable();
        }

        public static IEnumerable<string> GetAllNamedEntities(string[]
tokens)
        {
            var res = new List<string>();
            res.AddRange(GetNamedEntities(tokens, EntityType.Location));
            res.AddRange(GetNamedEntities(tokens, EntityType.Money));
            res.AddRange(GetNamedEntities(tokens,
EntityType.Organization));
            res.AddRange(GetNamedEntities(tokens, EntityType.Person));
            return res;
        }
    }
}



namespace NewsClassification.Preprocessing
{
    using System.Collections.Generic;
    using java.io;
    using opennlp.tools.sentdetect;

    public static class Sentenizer
    {
        private const string _modelFile = "en-sent.bin";
```

```
        private static readonly FileInputStream ModelIn;
        private static readonly SentenceModel Model;

        static Sentenizer()
        {

            ModelIn = new FileInputStream(_modelFile);
            Model = new SentenceModel(ModelIn);
        }

        public static IEnumerable<string> ExtractSentences(string contents)
        {
            SentenceDetectorME sentenizer = new SentenceDetectorME(Model);
            string[] sentences = sentenizer.sentDetect(contents);

            return sentences;
        }
    }
}



namespace NewsClassification.Preprocessing
{
    using System.Collections.Generic;
    using System.Text;
    using java.io;
    using opennlp.tools.tokenize;

    public static class Tokenizer
    {
        private const string _modelFile = "en-token.bin";

        private static readonly FileInputStream ModelIn;
        private static readonly TokenizerModel Model;

        static Tokenizer()
        {
            ModelIn = new FileInputStream(_modelFile);
            Model = new TokenizerModel(ModelIn);
        }
        public static IEnumerable<string> TokenizeNow(string contents)
        {
            string processedContents = PreProcessing(contents);
            TokenizerME tokenizer = new TokenizerME(Model);
            string[] tokens = tokenizer.tokenize(processedContents);
            return tokens;
        }

        private static string PreProcessing(string inp)
        {
            StringBuilder sb = new StringBuilder();

            //Retain only characters
            foreach (char c in inp)
            {
                if (c >= 'A' && c <= 'Z' || c >= 'a' && c <= 'z')
                {
                    sb.Append(c);
                }
                else
                {
                    sb.Append(' ');
```

```
                }
            }

            return sb.ToString();
        }
    }
}



namespace NewsClassification.ResultAnalyzerStrategy
{
    using System.Collections.Generic;

    public interface IResultAnalyzer
    {
        List<string> AnalyzeResults(IDictionary<string, double> results);
    }
}



namespace NewsClassification.ResultAnalyzerStrategy
{
    using System.Collections.Generic;
    using System.Linq;

    public class MinResultAnalyzer : IResultAnalyzer
    {
        private double _minValue;

        public MinResultAnalyzer(double minValue)
        {
            _minValue = minValue;
        }

        public List<string> AnalyzeResults(IDictionary<string, double>
results)
        {
            return results.Where(x => x.Value > _minValue).Select(x =>
x.Key).ToList();
        }
    }
}



namespace NewsClassification.ResultMergeStrategy
{
    using System.Collections.Generic;

    public interface IResultMergeStrategy
    {
        Dictionary<string, double> MergeResults(List<IDictionary<string,
double>> results);
        Dictionary<string, double> MergeResults(params IDictionary<string,
double>[] results);
    }
}
```

```
namespace NewsClassification.ResultMergeStrategy
{
    using System.Collections.Generic;
    using System.Linq;

    public class NaiveStrategy : IResultMergeStrategy
    {
        public Dictionary<string, double>
MergeResults(List<IDictionary<string, double>> results)
        {
            var mergedResult = new Dictionary<string, double>();

            foreach (IDictionary<string, double> result in results)
            {
                foreach (KeyValuePair<string, double> resultKV in result)
                {
                    if (!mergedResult.ContainsKey(resultKV.Key))
                    {
                        mergedResult[resultKV.Key] = resultKV.Value;
                    }
                    else
                    {

                        mergedResult[resultKV.Key] += resultKV.Value;
                    }
                }
            }

            foreach (KeyValuePair<string, double> kv in mergedResult)
            {
                mergedResult[kv.Key] /= results.Count;
            }

            return mergedResult;
        }

        public Dictionary<string, double> MergeResults(params
IDictionary<string, double>[] results)
        {
            return MergeResults(results.ToList());
        }
    }
}
```

```
namespace NewsClassification
{
    using System.Collections.Generic;
    using System.Linq;
    using Classifier;
    using NER;
    using Preprocessing;
    using ResultAnalyzerStrategy;
    using ResultMergeStrategy;

    public class NewsClassifier
    {
        private IClassifier _textBasedClassifier;
        private IClassifier _nerBasedClassifier;
        private bool _isNerClassifierEnabled;
```

```
        private IResultMergeStrategy _mergeStrategy;
        private IResultAnalyzer _resultAnalyzer;

        public IClassifier TextBasedClassifier
        {
            get => _textBasedClassifier;
            set => _textBasedClassifier = value;
        }

        public IClassifier NerBasedClassifier
        {
            get => _nerBasedClassifier;
            set => _nerBasedClassifier = value;
        }

        public bool IsNerClassifierEnabled
        {
            get => _isNerClassifierEnabled;
            set => _isNerClassifierEnabled = value;
        }

        public IResultMergeStrategy MergeStrategy
        {
            get => _mergeStrategy;
            set => _mergeStrategy = value;
        }

        public IResultAnalyzer ResultAnalyzer
        {
            get => _resultAnalyzer;
            set => _resultAnalyzer = value;
        }

        public List<string> ClassifyNewsArticle(string text)
        {
            string[] words = Tokenizer.TokenizeNow(text).ToArray();
            string[] namedEntities =
NamedEntitiesRecognizer.GetAllNamedEntities(words).ToArray();

            IDictionary<string, double> fullResult;
            IDictionary<string, double> textResult =
_textBasedClassifier.Classify(words);

            if (namedEntities.Any() && _isNerClassifierEnabled)
            {
                IDictionary<string, double> nerResult =
_nerBasedClassifier.Classify(namedEntities);
                fullResult = _mergeStrategy.MergeResults(textResult,
nerResult);
            }
            else
            {
                fullResult = textResult;
            }

            return _resultAnalyzer.AnalyzeResults(fullResult);
        }
    }
}
```

```
namespace Sample
{
    using System;
    using System.Collections.Generic;
    using NewsClassification.Builder;

    class Program
    {
        static void Main(string[] args)
        {
            Director director = new Director();
            var naiveBuilder = new NaiveBuilder();
            director.Construct(naiveBuilder);

            var newsClassifier = naiveBuilder.GetResult();

            const string newsText = "All the countries President Obama has
visited in the last 8 years mapped";
            List<string> classes =
newsClassifier.ClassifyNewsArticle(newsText);

            Console.WriteLine($"{newsText} is classified as {string.Join(",
", classes)}");
        }
    }
}
```

Додаток 2

Копія презентації